

Technisches

Referenzhandbuch

APCI-3600

Geräusch- und Vibrationsmesskarte, galvanisch getrennt



Produktinformation

Dieses Handbuch enthält die technischen Anlagen, wichtige Anleitungen zur korrekten Inbetriebnahme und Nutzung sowie Produktinformation entsprechend dem aktuellen Stand vor der Drucklegung.

Der Inhalt dieses Handbuchs und die technischen Daten des Produkts können ohne vorherige Ankündigung geändert werden. Die ADDI-DATA GmbH behält sich das Recht vor, Änderungen bzgl. der technischen Daten und der hierin enthaltenen Materialien vorzunehmen.

Gewährleistung und Haftung

Der Nutzer ist nicht berechtigt, über die vorgesehene Nutzung des Produkts hinaus Änderungen des Werks vorzunehmen sowie in sonstiger Form in das Werk einzugreifen.

ADDI-DATA übernimmt keine Haftung bei offensichtlichen Druck- und Satzfehlern. Darüber hinaus übernimmt ADDI-DATA, soweit gesetzlich zulässig, weiterhin keine Haftung für Personen- und Sachschäden, die darauf zurückzuführen sind, dass der Nutzer das Produkt unsachgemäß installiert und/oder in Betrieb genommen oder bestimmungswidrig verwendet hat, etwa indem das Produkt trotz nicht funktionsfähiger Sicherheits- und Schutzvorrichtungen betrieben wird oder Hinweise in der Betriebsanleitung bzgl. Transport, Lagerung, Einbau, Inbetriebnahme, Betrieb, Grenzwerte usw. nicht beachtet werden. Die Haftung ist ferner ausgeschlossen, wenn der Betreiber das Produkt oder die Quellcode-Dateien unbefugt verändert und/oder die ständige Funktionsbereitschaft von Verschleißteilen vorwerfbar nicht überwacht wurde und dies zu einem Schaden geführt hat.

Urheberrecht

Dieses Handbuch, das nur für den Betreiber und dessen Personal bestimmt ist, ist urheberrechtlich geschützt. Die in der Betriebsanleitung und der sonstigen Produktinformation enthaltenen Hinweise dürfen vom Nutzer des Handbuchs weder vervielfältigt noch verbreitet und/oder Dritten zur Nutzung überlassen werden, soweit nicht die Rechtsübertragung im Rahmen der eingeräumten Produktlizenz gestattet ist. Zuwiderhandlungen können zivil- und strafrechtliche Folgen nach sich ziehen.

ADDI-DATA-Software-Produktlizenz

Bitte lesen Sie diese Lizenz sorgfältig durch, bevor Sie die Standardsoftware verwenden.

Das Recht zur Benutzung dieser Software wird dem Kunden nur dann gewährt, wenn er den Bedingungen dieser Lizenz zustimmt.

Die Software darf nur zur Einstellung der ADDI-DATA-Produkte verwendet werden.

Das Kopieren der Software ist verboten (außer zur Archivierung/Datensicherung und zum Austausch defekter Datenträger). Deassemblierung, Dekompilierung, Entschlüsselung und Reverse Engineering der Software ist verboten. Diese Lizenz und die Software können an eine dritte Partei übertragen werden, sofern diese Partei ein Produkt käuflich erworben hat, sich mit allen Bestimmungen in diesem Lizenzvertrag einverstanden erklärt und der ursprüngliche Besitzer keine Kopien der Software zurückhält.

Warenzeichen

- ADDI-DATA, APCI-1500, MSX-Box und MSX-E sind eingetragene Warenzeichen der ADDI-DATA GmbH.
- Turbo Pascal, Delphi, Borland C, Borland C++ sind eingetragene Warenzeichen der Borland Software Corporation.
- Microsoft .NET, Microsoft C, Visual C++, MS-DOS, Windows 95, Windows 98, Windows 2000, Windows NT, Windows EmbeddedNT, Windows XP, Windows Vista, Windows 7, Windows Server 2000, Windows Server 2003, Windows Embedded und Internet Explorer sind eingetragene Warenzeichen der Microsoft Corporation.
- LabVIEW, LabWindows/CVI, DASyLab, DIAdem sind eingetragene Warenzeichen der National Instruments Corporation.
- CompactPCI ist ein eingetragenes Warenzeichen der PCI Industrial Computer Manufacturers Group.
- VxWorks ist ein eingetragenes Warenzeichen von Wind River Systems, Inc.
- RTX ist ein eingetragenes Warenzeichen von IntervalZero.

Warnung

Bei unsachgemäßem Einsatz und bestimmungswidrigem Gebrauch der Karte können:



Personen verletzt werden



Karte, PC und Peripherie beschädigt werden



Umwelt verunreinigt werden

■ Schützen Sie sich, andere und die Umwelt!

■ Lesen Sie unbedingt die Sicherheitshinweise!

Liegen Ihnen keine Sicherheitshinweise vor, so fordern Sie diese bitte an.

■ Beachten Sie die Anweisungen des Handbuchs!

Vergewissern Sie sich, dass Sie keinen Schritt vergessen haben.

Wir übernehmen keine Verantwortung für Schäden, die aus dem falschen Einsatz der Karte hervorgehen könnten.

■ Beachten Sie folgende Symbole:



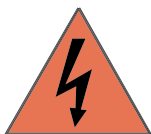
WICHTIG!

Kennzeichnet Anwendungstipps und andere nützliche Informationen.



WARNUNG!

Bezeichnet eine möglicherweise gefährliche Situation. Bei Nichtbeachten des Hinweises können Karte, PC und/oder Peripherie **zerstört** werden.



WARNUNG!

Bezeichnet eine möglicherweise gefährliche Situation. Bei Nichtbeachten des Hinweises können Karte, PC und/oder Peripherie **zerstört** und Personen **gefährdet** werden.

1	DEFINITION DES VERWENDUNGSBEREICHS	11
1.1	Bestimmungsgemäßer Zweck	11
1.2	Bestimmungswidriger Zweck	11
1.3	Grenzen der Verwendung	11
1.4	Allgemeine Beschreibung der Karte	12
2	BENUTZER	13
2.1	Qualifikation	13
2.2	Länderspezifische Bestimmungen	13
3	HANDHABUNG DER KARTE	14
4	TECHNISCHE DATEN	15
4.1	Elektromagnetische Verträglichkeit (EMV)	15
4.2	Mechanischer Aufbau	15
4.3	Versionen	16
4.4	Grenzwerte	17
4.4.1	Analoge Eingänge	18
4.4.2	Analoge Ausgänge	20
4.4.3	Digitale Eingänge	21
4.4.4	Digitale Ausgänge	21
4.4.5	Stromquellen	21
4.4.6	Chronometer	22
4.4.7	Externer Clock 1/2 und Mastertrigger	22
5	EINBAU DER KARTE	23
5.1	PC öffnen	23
5.2	Auswahl eines freien Steckplatzes	23
5.3	Einbau	24
5.4	PC schließen	24
6	SOFTWARE	25
6.1	Kartenkonfiguration mit ADDIREG	25
6.1.1	Installationsbeschreibung	25
6.1.2	Die Schaltfläche "More information"	29
6.1.3	PCI analoge Eingabekarten mit DMA	29
6.1.4	Eine neue Karte registrieren	33
6.1.5	Die Registrierung einer vorhandenen Karte ändern	33
6.2	Fragen und Software-Download im Internet	34

7	ANSCHLUSS AN DIE PERIPHERIE	35
7.1	Steckerbelegungen.....	35
7.2	Versionsverwaltung	38
8	FUNKTIONEN DER KARTE	39
8.1	Überblick.....	39
8.2	Blockschaltbild.....	39
8.3	Analoge Eingänge.....	40
8.3.1	Verteilung der analogen Eingänge	40
8.3.2	Coupling Mode	41
8.3.3	Konfiguration	41
8.3.4	Eingangsbereich	41
8.3.5	Kalibrierung.....	41
8.3.6	Anti-Aliasing-Filter	41
8.3.7	A/D-Wandler	46
8.3.8	Stromquellen	48
8.3.9	Modes	49
8.4	Analoge Ausgänge.....	50
8.4.1	Modes der analogen Ausgänge	51
8.5	Chronometer-Eingänge.....	52
8.5.1	Modes der Chronometer.....	54
8.6	Digitale Eingänge.....	56
8.7	Digitale Ausgänge.....	57
8.8	Externe Clocks und Mastertrigger.....	57
8.8.1	Master Mode	59
8.8.2	Slave Mode	60
8.9	On Board-Speicher (SDRAM)	61
8.10	Speicherkonzept	61
8.10.1	Speicherkonzept: Globaler Speicher	61
8.10.2	Speicherkonzept: Ring Buffer	62
8.11	Maximale Datenübertragungsgeschwindigkeit	64
9	SOFTWARE	65
9.1	Softwarefunktionen	65
9.1.1	Allgemeine Funktionen	65
1)	i_PCI3600_InitCompiler()	65
2)	i_PCI3600_GetBoardList ()	66
3)	i_PCI3600_OpenBoard ()	67
4)	i_PCI3600_InitBoard ()	68
5)	i_PCI3600_GetBoardInitialisation ().....	71

6)	i_PCI3600_GetBoardInformation ()	73
7)	i_PCI3600_CloseBoard ()	74
8)	i_PCI3600_SetBoardIntRoutine ()	75
9)	i_PCI3600_ResetBoardIntRoutine ()	83
10)	i_PCI3600_Write32BitPortValue()	84
11)	i_PCI3600_Read32BitPortValue()	85
9.1.2	Analoge Eingänge	86
12)	i_PCI3600_InitAnalogInputModule()	86
13)	i_PCI3600_ReleaseAnalogInputModule()	89
14)	i_PCI3600_GetAnalogInputModuleCalibrationStatus()	90
15)	i_PCI3600_StartAnalogInputModuleSingleAcquisition()	91
16)	i_PCI3600_GetAnalogInputModuleSingleAcquisitionStatus()	92
17)	i_PCI3600_ReadAnalogInputSingleAcquisitionValues()	93
18)	i_PCI3600_GetAnalogInputModuleValues()	94
19)	i_PCI3600_InitAnalogInputModuleCyclicAcquisition()	95
20)	i_PCI3600_StartAnalogInputModuleCyclicAcquisition()	97
21)	i_PCI3600_GetAnalogInputModuleCyclicAcquisitionStatus()	98
22)	i_PCI3600_IncrementAnalogInputCyclicAcquisitionBufferReadPointer()	99
23)	i_PCI3600_StopAnalogInputModuleCyclicAcquisition()	100
24)	i_PCI3600_ReleaseAnalogInputModuleCyclicAcquisition()	101
25)	i_PCI3600_AnalogInput_EnableDisableDigitalInputExternalTrigger()	102
9.1.3	Analoge Ausgänge	103
26)	i_PCI3600_InitAnalogOutputChannel()	103
27)	i_PCI3600_ReleaseAnalogOutputChannel()	105
28)	i_PCI3600_GetAnalogOutputReadyBitStatus()	106
29)	i_PCI3600_WriteAnalogOutputValue()	107
30)	i_PCI3600_Set1AnalogOutputChannel()	108
31)	i_PCI3600_ReserveAnalogOutputSignalGeneratorComputerBuffer()	109
32)	i_PCI3600_FreeAnalogOutputSignalGeneratorComputerBuffer()	110
33)	i_PCI3600_WriteAnalogOutputSignalGeneratorComputerBufferValues()	111
34)	i_PCI3600_InitAnalogOutputSignalGenerator()	113
35)	i_PCI3600_StartAnalogOutputSignalGenerator()	115
36)	i_PCI3600_GetAnalogOutputSignalGeneratorStatus()	116
37)	i_PCI3600_StopAnalogOutputSignalGenerator()	118
38)	i_PCI3600_ReleaseAnalogOutputSignalGenerator()	119

	39)	i_PCI3600_AnalogOutput_EnableDisableDigitalInputExternalTrigger()	120
9.1.4		Chronometer-Modul.....	121
	40)	i_PCI3600_InitChronometerModule()	121
	41)	i_PCI3600_ReleaseChronometerModule()	122
	42)	i_PCI3600_StartChronometerModuleAcquisition()	123
	43)	i_PCI3600_GetChronometerModuleFIFOStatus()	124
	44)	i_PCI3600_ReadChronometerModuleValue()	125
	45)	i_PCI3600_StopChronometerModuleAcquisition()	126
	46)	i_PCI3600_InitChronometerModuleCyclicAcquisition	127
	47)	i_PCI3600_StartChronometerModuleCyclicAcquisition()...	129
	48)	i_PCI3600_GetChronometerModuleCyclicAcquisitionStatus()	130
	49)	i_PCI3600_IncrementChronometerCyclicAcquisitionBufferReadPointer()	131
	50)	i_PCI3600_StopChronometerModuleCyclicAcquisition() ..	132
	51)	i_PCI3600_ReleaseChronometerModuleCyclicAcquisition()	133
	52)	i_PCI3600_Chronometer_EnableDisableDigitalInputExternalTrigger().....	134
9.1.5		Digitale Eingänge	135
	53)	i_PCI3600_Read1DigitalInput()	135
	54)	i_PCI3600_Read4DigitalInputs()	136
	55)	i_PCI3600_Read8DigitalInputs()	137
9.1.6		Digitale Ausgänge	138
	56)	i_PCI3600_EnableDisableDigitalOutputMemory()	138
	57)	i_PCI3600_Set1DigitalOutputOn().....	139
	58)	i_PCI3600_Set1DigitalOutputOff().....	140
	59)	i_PCI3600_Set4DigitalOutputsOn()	141
	60)	i_PCI3600_Set4DigitalOutputsOff()	142
	61)	i_PCI3600_Set8DigitalOutputsOn()	143
	62)	i_PCI3600_Set8DigitalOutputsOff()	144
9.1.7		Externer Trigger	145
	63)	i_PCI3600_InitDigitalInputExternalTrigger()	145
9.1.8		Mastertrigger	146
	64)	_INT_ i_PCI3600_AnalogInput_EnableDisableMasterTrigger () :	146
	65)	_INT_ i_PCI3600_AnalogOutput_EnableDisableMasterTrigger () :	147
	66)	_INT_ i_PCI3600_Chronometer_EnableDisableMasterTrigger () :	148
	67)	_INT_ i_PCI3600_MasterTrigger () :	149

10 ANHANG 150

10.1 Verwendete Abkürzungen 150

10.2	Glossar	151
11	INDEX	154
11.1	154	

Abbildungen

Abb. 3-1: Richtige Handhabung	14
Abb. 4-1: Revision C (Rückseite der Karte)	16
Abb. 4-2: Benötigte Steckplätze	17
Abb. 5-1: Steckplatztypen	23
Abb. 5-2: Einbau der Karte	24
Abb. 5-3: Die Karte an der Gehäuserückwand befestigen	24
Abb. 6-1: ADDIREG-Hauptfenster (Beispiel)	26
Abb. 6-2: Eine neue Karte einbauen	28
Abb. 6-3: Verwaltung der PCI DMA-Karten	30
Abb. 7-1: Koaxialer SMB-Stiftstecker auf Frontblech	35
Abb. 7-2: Digitale Ein- und Ausgänge: 26-pol. Pfostenstecker auf 37-pol. SUB-D-Stiftstecker (zweites Frontblech: Flachbandkabel FB3600-D)	36
Abb. 7-3: Analoge Ausgänge: 14-pol. Pfostenstecker auf 15-pol. SUB-D Buchsenstecker Chronometer-Eingänge: 14-pol. Pfostenstecker auf 15-pol. SUB-D Stiftstecker (zweites Frontblech: Flachbandkabel FB3600-AC)	37
Abb. 7-4: Externe Clock: Pfostenstecker	37
Abb. 8-1: Blockschaltbild der APCI-3600	39
Abb. 8-2: Eingangsschaltung eines analogen Eingangs	40
Abb. 8-3: Aliasing-Effekt: Zeitlicher Aspekt	42
Abb. 8-4: Abtastung von Analogsignalen	43
Abb. 8-5: Single Speed Mode (Stoband-Dämpfung)	45
Abb. 8-6: Single Speed Mode (Transition-Band)	45
Abb. 8-7: Double Speed Mode (Stopband-Dämpfung)	45
Abb. 8-8: Double Speed Mode (Transition-Band)	45
Abb. 8-9: Quad Speed Mode (Stopband- Dämpfung)	45
Abb. 8-10: Quad Speed Mode (Transition-Band)	45
Abb. 8-11: ICP-Sensor Versorgung mit der APCI-3600	49
Abb. 8-12: Chronometer-Eingänge	53
Abb. 8-13: Digitale Eingänge	56
Abb. 8-14: Digitale Ausgänge	57
Abb. 8-15: Karte im Master Mode	60
Abb. 8-16: Karte im Slave Mode	61
Abb. 8-17: Globaler Speicher	62
Abb. 8-18: Ring Buffer	63
Abb. 9-1: Synchroner und asynchroner Mode	77

Tabellen

Tabelle 4-1: Auswählbare Frequenzen	18
Tabelle 8-1: Kanäle und A/D-Wandler	40
Tabelle 8-2: Anti-Aliasingfilter: Eingangsstufen	44
Tabelle 8-3: Abtast-Frequenzbereich	47
Tabelle 8-4: Abtastfrequenz der analogen Eingänge	48
Tabelle 8-5: Abtastfrequenz	51
Tabelle 8-6: Abtastfrequenz der analogen Ausgänge	52
Tabelle 8-7: Clock-Generator Werte.....	59
Tabelle 8-8: Datenübertragungsgeschwindigkeit	64
Tabelle 9-1: Clock-Divisor	69
Tabelle 9-2: Clock-Auswahl	69
Tabelle 9-3: Synchroner Mode	77
Tabelle 9-4: Asynchroner Mode	78
Tabelle 9-5: dw - Interruptquelle.....	79
Tabelle 9-6: dw - Interruptmaske	79
Tabelle 9-7: Speicher	80
Tabelle 9-8: ADC-Clocks	87
Tabelle 9-9: RAM-Speichergröße auf der Karte	96
Tabelle 9-10: DAC Sampling-Clock.....	103

1 DEFINITION DES VERWENDUNGSBEREICHS

1.1 Bestimmungsgemäßer Zweck

Die Karte **APCI-3600** eignet sich für den Einbau in einen PC mit PCI 3,3 V/64-Bit Steckplätzen, der für die elektrische Mess-, Steuer-, Regel- und Labortechnik im Sinne der EN 61010-1 (IEC 61010-1) eingesetzt wird.

Der verwendete Personal Computer (PC) muss die Anforderungen von IEC 60950-1 oder EN 60950-1 und EN 55022 oder IEC/CISPR 22 und EN 55024 oder IEC/CISPR 24 erfüllen.

Der Einsatz der Karte **APCI-3600** in Kombination mit externen Anschlussplatinen setzt eine fachgerechte Installation nach IEC 60439-1 oder EN 60439-1 (Schaltschrank/Schaltkasten) voraus.

1.2 Bestimmungswidriger Zweck

Die Karte **APCI-3600** darf nicht als sicherheitsbezogenes Betriebsmittel (safety related part, SRP) eingesetzt werden.

Es dürfen keine sicherheitsbezogenen Funktionen, wie beispielsweise NOT-AUS-Einrichtungen gesteuert werden.

Die Karte **APCI-3600** darf nicht in explosionsgefährdeten Atmosphären eingesetzt werden.

Die Karte **APCI-3600** darf nicht als elektrisches Betriebsmittel im Sinne der Niederspannungsrichtlinie 2006/95/EG betrieben werden.

1.3 Grenzen der Verwendung

Die bestimmungsgemäße Verwendung erfordert das Beachten aller Sicherheitshinweise und des technischen Referenzhandbuchs.

Eine andere oder darüber hinausgehende Benutzung gilt als nicht bestimmungsgemäß. Für hieraus entstehende Schäden haftet der Hersteller nicht.

Die Karte muss bis zum Einsatz in ihrer Schutzverpackung bleiben.

Entfernen Sie nicht die Kennzeichnungsnummern der Karte, da dadurch ein Garantieverlust entsteht.

1.4 Allgemeine Beschreibung der Karte

Die Erfassung analoger Daten mit der Karte **APCI-3600** erfolgt über koaxiale Kabel, die an die SMB-koaxialen Stiftstecker der Karte **APCI-3600** anzuschließen sind, während die analogen Ausgänge an einen SUB-D-Stiftstecker anzuschließen sind.

Der Anschluss unseres Standardkabels **ST3600** erfüllt die Mindestforderungen:

- metallisierte Steckergehäuse
- geschirmtes Kabel

Ihre Karte besitzt folgende Merkmale:

APCI-3600:

- 8 (4x2) analoge Eingänge (Kapitel 8.3)
- 2 analoge Ausgänge (Kapitel 8.4)
- 4 Chronometer-Eingänge (Kapitel 8.5)
- 8 digitale Eingänge (Kapitel 8.6)
- 8 digitale Ausgänge (Kapitel 8.7)
- 2 externe Clocks (Kapitel 8.8)
- On Board SDRAM (Kapitel 8.9)

APCI-3600-L:

- 8 (4x2) analoge Eingänge (Kapitel 8.3)
- On Board SDRAM (Kapitel 8.9)

2 BENUTZER

2.1 Qualifikation

Nur eine ausgebildete Elektronikfachkraft darf folgende Tätigkeiten ausführen:

- Installation
- Inbetriebnahme
- Betrieb
- Instandhaltung.

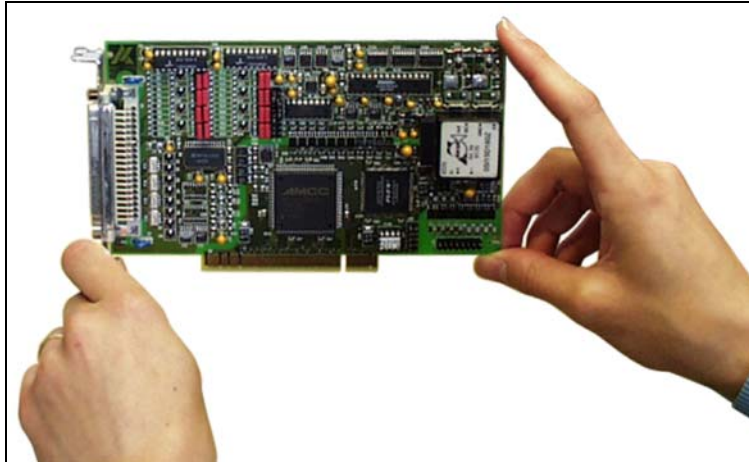
2.2 Länderspezifische Bestimmungen

Beachten Sie die länderspezifischen Bestimmungen zur:

- Unfallverhütung
- Errichtung von elektrischen und mechanischen Anlagen
- Elektromagnetische Verträglichkeit (EMV).

3 HANDHABUNG DER KARTE

Abb. 3-1: Richtige Handhabung



4 TECHNISCHE DATEN

4.1 Elektromagnetische Verträglichkeit (EMV)

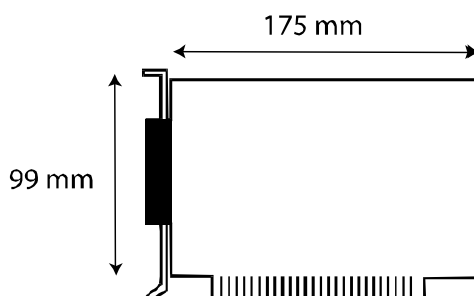
Die Karte **APCI-3600** entspricht den Anforderungen der europäischen EMV-Richtlinie. Die Prüfungen wurden nach der zutreffenden Norm aus der EN 61326 – Reihe (IEC 61326) von einem akkreditierten EMV-Labor durchgeführt. Die Grenzwerte werden im Sinne der europäischen EMV-Richtlinie für eine industrielle Umgebung eingehalten.

Der entsprechende EMV-Prüfbericht kann angefordert werden.

4.2 Mechanischer Aufbau

Die Karte ist auf einer 6-Lagen Leiterplatte aufgebaut.

Abmessungen:



Gewicht: ca. 170 g

Einbau in: 32/64-Bit PCI Steckplatz, 5 V

Anschluss zur Peripherie:

Frontstecker: 8 x SMB koaxiale Stiftstecker

Zusätzliche Stecker: 3 zusätzliche Frontblechöffnungen:

- 37-pol. SUB-D Stiftstecker

- 2 x 15-pol. SUB-D Stiftstecker

- 6-pol. Stiftstecker

Zubehör¹:

ST3601: Verbindungskabel RG 178, 2 m
SMB koaxialer Buchsenstecker auf
BNC-Stiftstecker

Flachbandkabel:

Flachbandkabel **FB3600-D:** Für Anschluss der digitalen E/A:
von 26- pol. Pfostenstecker auf 37-pol
SUB-D Stiftstecker

Flachbandkabel **FB3600-AC:** 1) Für Anschluss der analogen

¹ Nicht im Standard-Lieferumfang enthalten.

Ausgänge:
 von 14-pol. Pfostenstecker auf 15-pol.
 SUB-D Stiftstecker
 2) Für Anschluss des Chronometers:
 von 14- pol. Pfostenstecker auf 15-pol.
 SUB-D Stiftstecker



WARNUNG!

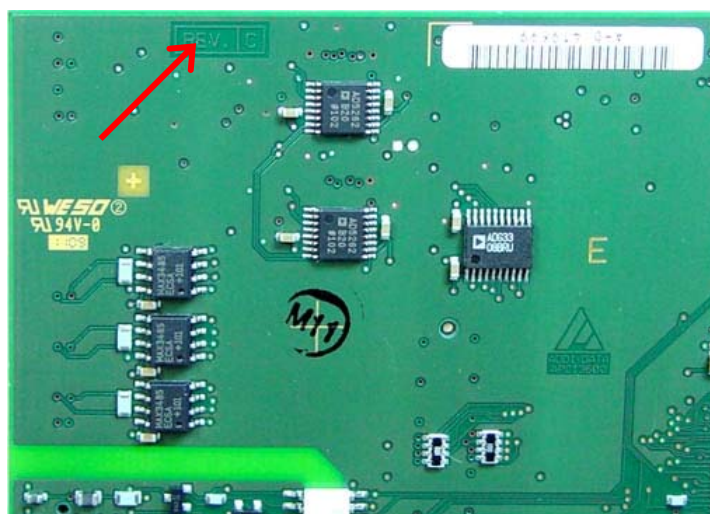
Die Anschlussleitungen sind gegen mechanische Belastung zu verlegen.

4.3 Versionen

Die Karte ist in 2 Versionen erhältlich:

- | | |
|--------------------|---|
| APCI-3600: | <ul style="list-style-type: none"> - 8 analoge Eingängen - 4 bzw. 8 Stromquellen (ab Revision C, siehe Abb. 4-1) zum Anschluss von ICP-Sensoren - 2 analoge Ausgänge - 4 Chronometer-Eingänge - 8 digitale Eingänge - 8 digitale Ausgänge - 128 MBytes SDRAM |
| APCI-3600-L: | <ul style="list-style-type: none"> - 8 analoge Eingänge - 4 bzw. 8 Stromquellen (ab Revision C, siehe Abb. 4-1) zum Anschluss von ICP-Sensoren - 128 MBytes SDRAM |

Abb. 4-1: Revision C (Rückseite der Karte)



4.4 Grenzwerte

Höhenlage: 2000 m über NN

Betriebstemperatur: 0 bis 60°C

Lagertemperatur: -25 bis + 70°C

Relative Luftfeuchtigkeit bei Innenraumaufstellung:

50% bei +40 °C

80% bei +31 °C

PC-Mindestvoraussetzungen:

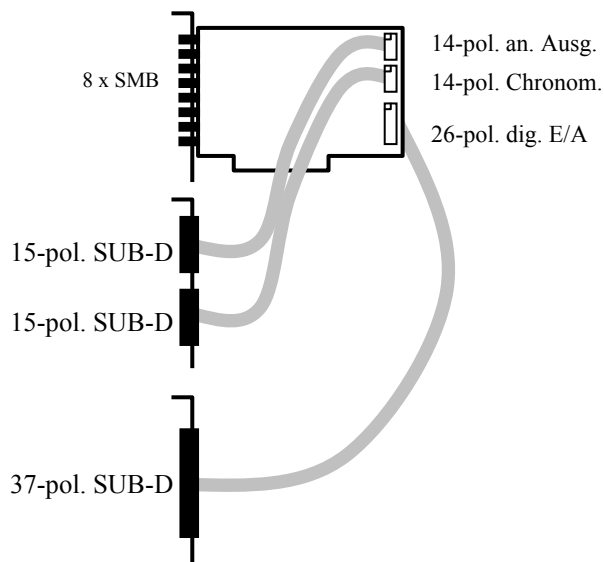
PCI BIOS ab Version 1.0

Bus Geschwindigkeit: < 33 MHz

Betriebssystem: Windows Vista (32-Bit), XP, 2000,
Linux

Steckplätze: 1) PCI 5 V oder PCI 3,3 V, 32-Bit
2) Steckplatzöffnung für analoge
Ausgänge und Chronometer-Eingänge
3) Steckplatzöffnung für Anschluss der
digitalen E/A

Abb. 4-2: Benötigte Steckplätze



Energiebedarf:

- Betriebsspannung vom PC: 5 V $\pm$ 5%

- Stromverbrauch (ohne Last): Siehe Tabelle ($\pm$ 5%)

	APCI-3600	APCI-3600-L
+ 5 V vom PC	1,75 A	1,75 A

4.4.1 Analoge Eingänge

Anzahl:	8
Eingangstyp:	Single-Ended oder differenziell durch Software auswählbar
Auflösung:	24-Bit
A/D Wandler:	Delta-Sigma, 5. Ordnung, Multibit-Delta-Sigma Modulator
Verstärkung:	x1, x10
Eingangsbereiche:	± 10 V Single-Ended (Verstärkung x1) ± 1 V Single-Ended (Verstärkung x10) ± 5 V differentiell (Verstärkung x1) ± 0,5 V differentiell (Verstärkung x10)
Abtastfrequenz f_s :	$2 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$ durch Software auswählbar; (Siehe untere Tabelle)

Tabelle 4-1: Auswählbare Frequenzen

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$	$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$	$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$
50000 Hz	100000 Hz	200000 Hz
40000 Hz	80000 Hz	160000 Hz
33333 Hz	66667 Hz	133333 Hz
25000 Hz	50000 Hz	100000 Hz
20000 Hz		
16667 Hz		
12500 Hz		
10000 Hz		
8000 Hz		
5000 Hz		
4000 Hz		
3333 Hz		
2500 Hz		
2000 Hz		

Oversampling:	$64 \times f_s$ (für Abtastfrequenz f_s)
Frequenzgenauigkeit:	± 50 ppm
FIFO-Tiefe:	128 DWORD, für rechten und linken Kanal desselben ADC
Datentransfer:	DMA, I/O, IRQ

Durchlassbereich-Ripple (rel. zu 1 kHz), max., DC-gekoppelt:

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$	-0,1dB, DC bis $0,47 \times f_s$
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$	-0,1dB, DC bis $0,45 \times f_s$
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$	-0,1dB, DC bis $0,24 \times f_s$

-3 dB Bandbreite:

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$	$0,5 \times f_s$
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$	$0,5 \times f_s$
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$	$0,358 \times f_s$

Eingangskopplung:	AC, DC, GND, durch Software auswählbar
AC -3dB Grenzfrequenz:	1,6 Hz

Überspannungsschutz

R0-, L0-, R1-, L1-, L/R2+-, L/R3+-

Maximaler Gleichstrom: $\pm 12 \text{ V}$, $\pm 200 \text{ mA}$

Maximaler Spitzenstrom

(Impuls bei 1ms, 10% duty cycle): $\pm 12 \text{ V}$, $\pm 300 \text{ mA}$

R0+, L0+, R1+, L1+

Maximaler Gleichstrom: $\pm 36 \text{ V}$, $\pm 30 \text{ mA}$

Maximaler Spitzenstrom

(Impuls bei 1ms, 10% duty cycle): $\pm 36 \text{ V}$, $\pm 70 \text{ mA}$

ESD-Schutz: $> 2 \text{ kV}$, ESD Schutz
durch Methode 3015.17

Dynamische Eigenschaften **$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:**Alias-freie Bandbreite (Passband)..... DC (0Hz) bis $0,47 \times f_s$, min. bis max.Stopband..... $0,58 \times f_s$ min

Stopband Dämpfung (Alias rejection): -95 dB min

Filter-Verzögerung durch ADC

(Total group delay): $12/f_s$ s typisch **$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:**Alias-freie Bandbreite (Passband)..... DC (0Hz) bis $0,45 \times f_s$, min. bis max.Stopband..... $0,68 \times f_s$ min

Stopband Dämpfung (Alias rejection): -92 dB min

Filter-Verzögerung durch ADC

(Total group delay): $9/f_s$ s typisch **$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:**Alias-freie Bandbreite (Passband)..... DC (0Hz) bis $0,24 \times f_s$, min. bis max.Stopband..... $0,78 \times f_s$ min

Stopband Dämpfung (Alias rejection): -97 dB min

Filter-Verzögerung durch ADC

(Total group delay): $5/f_s$ s typisch**Dynamikbereich****SNR:**

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:	$< -105 \text{ dB}$ (short input, Verstärkung x1)
	$< -100 \text{ dB}$ (short input, Verstärkung x10)
	$< -80 \text{ dB}$ (open input, Verstärkung x1)
	$< -60 \text{ dB}$ (open input, Verstärkung x10)

$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:	< -105 dB (short input, Verstärkung x1)
	< -100 dB (short input, Verstärkung x10)
	< -80 dB (open input, Verstärkung x1)
	< -60 dB (open input, Verstärkung x10)
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:	< -75 dB (short input, Verstärkung x1)
	< -75 dB (short input, Verstärkung x10)
	< -75 dB (open input, Verstärkung x1)
	< -60 dB (open input, Verstärkung x10)

Crosstalk (Übersprechen)(zwischen Kanal R0 und L0, R1 und L1, R2 und L2, R3 und L3 bei Verstärkung x1):

Short input bei $f_a = 100 \text{ Hz}$

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:	< -95 dB
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:	< -95 dB
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:	< -70 dB

Short input bei $f_a = 1 \text{ kHz}$:

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:	< -95 dB
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:	< -95 dB
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:	< -70 dB

1 k Ω Last bei $f_a = 100 \text{ Hz}$:

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:	< -95 dB
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:	< -95 dB
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:	< -70 dB

1 k Ω Last bei $f_a = 1 \text{ kHz}$:

$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$:	< -95 dB
$50 \text{ kHz} \leq f_s \leq 100 \text{ kHz}$:	< -95 dB
$100 \text{ kHz} \leq f_s \leq 200 \text{ kHz}$:	< -70 dB

Phasenfehler (zwischen Kanal R0 und L0, R1 und L1, R2 und L2, R3 und L3):

Bei $f_s = 200 \text{ kHz}$:	0,3° (max.)
	0,2° bei $f_a = 10 \text{ kHz}$ Sinussignal
	0,02° bei $f_a = 1 \text{ kHz}$ Sinussignal

Amplituden-Fehler: $\pm 0,02 \text{ dB}$, max. bei $f_s = 1 \text{ kHz}$
Sinussignal bei Verstärkung x1 und x10

Offset-Fehler: $\pm 200 \mu\text{V}$, max. bei $f_s = 2 \text{ kHz}$

4.4.2 Analoge Ausgänge

Anzahl der Ausgänge:	2
Auflösung:	16-Bit
Genauigkeit:	13-Bit
DAC Typ:	R-2R
Ausgangsbereich:	$\pm 10 \text{ V}$

Settling time:

10 V Step, $R_L = 2 \text{ k}\Omega$, $C_L = 1500 \text{ pF}$

$\pm 0,1\%$: 5 μs typisch

$\pm 0,01\%$: 5,6 μs typisch

Überspannungsschutz: $\pm 12 \text{ V}$, 100 mA max. Gleichstrom

Kurzschlussstrom: $\pm 45 \text{ mA}$ typisch

Ausgangsspannung nach Reset: 0 V

FIFO Tiefe: 256 Word

Datentransfer: DMA, IO, IRQ

4.4.3 Digitale Eingänge

Anzahl der Eingänge: 8

Filter/Schutzbeschaltung: Tiefpass/Transorbdiode

Galvanische Trennung: 1000 V

Nominalspannung: 24 V extern

Eingangsspannung: 0 bis 30 V

Eingangsstrom: 7 mA bei 24 VDC, typisch

Logische Eingangspegel:

UH (max): 30 V

UH (min): 19 V

UL (max): 14 V

UL (min): 0 V

Eingangsfrequenz (max.): 5 kHz bei 24 V

Trigger-Eingang: digitaler Eingang 0

4.4.4 Digitale Ausgänge

Anzahl Ausgänge: 8

Galvanische Trennung: 1000 V

Ausgangstyp: Open Kollektor

Nominalspannung: 24 V

Versorgungsspannung: 5-30 V

Ausgangsstrom pro Ausgang: 50 mA max.

Summenstrom: 300 mA begrenzt durch PTC

Anschaltzeit: 0,25 μs typisch

Ausschaltzeit: 0,25 μs typisch

4.4.5 Stromquellen

Anzahl: 4 bzw. 8 (ab Revision C, siehe Abb. 4-1)

Typ: Konstante Stromquelle für die Versorgung der ICP¹ Sensoren

Strom: 4 mA typisch

Standard: 24 V max.

¹ ICP Sensor: (integrated circuit piezoelectric)

4.4.6 Chronometer

Anzahl:	4 Chronometer, 2 Gate auf Chronometer 1 und 2
Eingangstyp:	RS485
Max. Geschwindigkeit:	1 MHz max.
Zählertiefe:	32-Bit
Divisor:	von 2^0 bis 2^{15} pro Zähler Eingang
FIFO Tiefe:	256 DWORD
Daten-Transfer:	DMA, IO, IRQ
Differential Threshold Voltage:	-200 mV min -50 mV max
Eingangswiderstand:	120 Ω differentiell
ESD-Schutz:	+/-15 kV Human Body Model

4.4.7 Externer Clock 1/2 und Mastertrigger

Typ:	differentiell, ANSI TIA/EIA-644-1995 Standard
Eingangsspannung:	
High-Pegel:	2 V min.
Low-Pegel:	0,8 V max.
Bereich der differentiellen Eingangsspannung, V_{ID}	0,1 V min. 0,6 V max.
Common-Mode Eingangsspannung:	$ V_{ID} /2$ min. 2,4 - $ V_{ID} /2$ max.

Receiver (Slave board)

Differentielle Eingangsspannung	
Positive Schaltschwelle:	50 mV min.
Negative Schaltschwelle:	-50 mV max.

Eingangstrom für jeden Eingangskanal: ...	-11 μ A typisch, $V_I = 0$ V -3 μ A typisch, $V_I = 2,4$ V
Eingangskapazität:	5 pF typisch

Driver (Master board)

Differentielle Eingangsspannung:	340 mV typisch
Änderung der differentiellen Eingangsspannung zwischen logischen Pegel	-50 mV min. 50 mV max.
Kurzschlussstrom:	10 mA max.
Ausgangstrom (Hohe Impedanz):	+/-1 μ A max.

5 EINBAU DER KARTE



WICHTIG!

Berücksichtigen Sie unbedingt die Sicherheitshinweise.

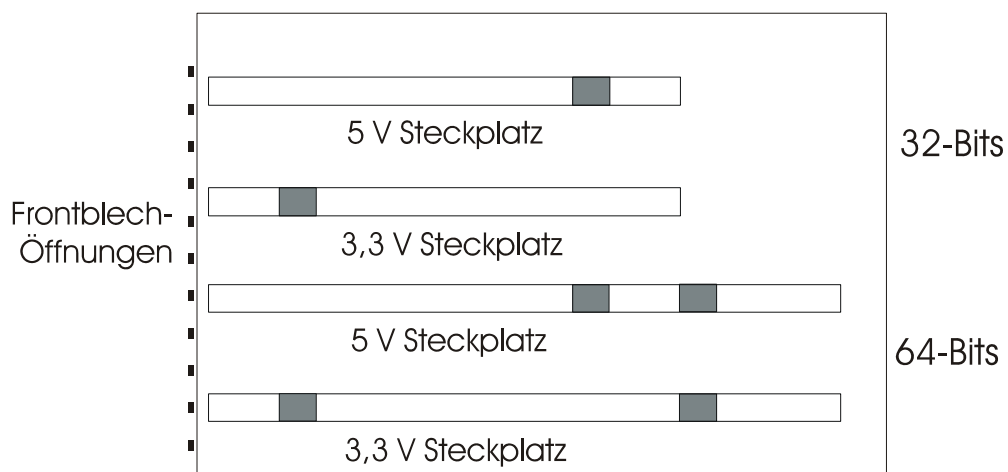
5.1 PC öffnen

- ♦ PC und alle am PC angeschlossenen Einheiten ausschalten.
- ♦ Netzstecker des PCs aus der Steckdose ziehen.
- ♦ PC öffnen wie im Handbuch des PC Herstellers beschrieben.

5.2 Auswahl eines freien Steckplatzes

- ♦ Stecken Sie die Karte in einen freien PCI-5V oder PCI-3,3 V (32/64-Bit) Steckplatz ein.

Abb. 5-1: Steckplatztypen



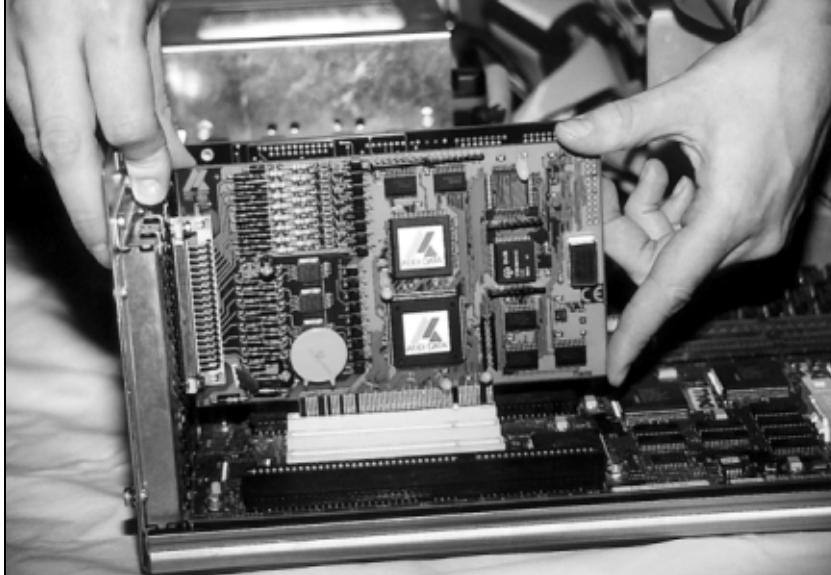
Das Blech des gewählten Steckplatzes ausschrauben. Bitte beachten Sie hierzu die Bedienungsanleitung des PC Herstellers. Bewahren Sie das Blech auf. Sie werden es nach dem eventuellen Ausbau der Karte wieder benötigen.

- ♦ Bitte sorgen Sie für einen Potentialausgleich.
- ♦ Entnehmen Sie die Karte aus ihrer Schutzverpackung.

5.3 Einbau

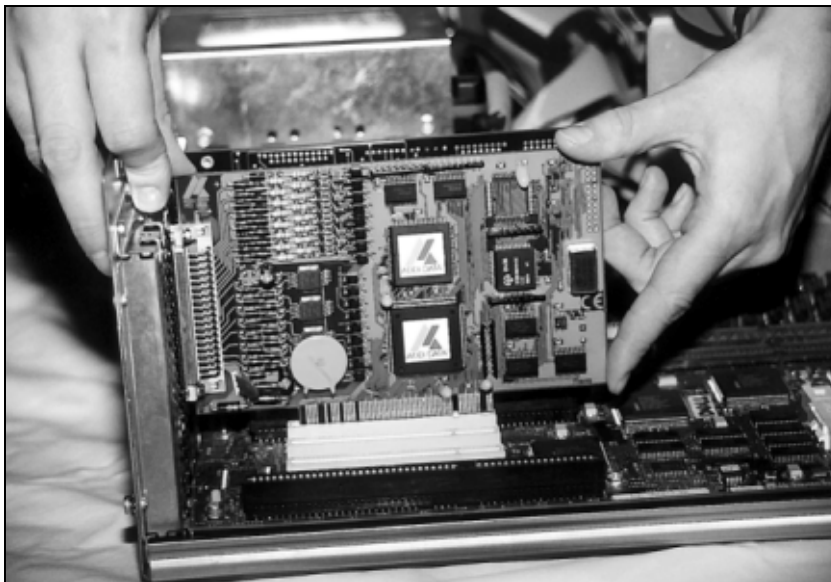
- ♦ Karte senkrecht von oben in den gewählten Steckplatz einführen.

Abb. 5-2: Einbau der Karte



- ♦ Karte an der Gehäuserückwand mit der Schraube befestigen, mit der das Blech befestigt war.

Abb. 5-3: Die Karte an der Gehäuserückwand befestigen



- ♦ Alle gelösten Schrauben festschrauben.

5.4 PC schließen

- ♦ PC schließen wie im Handbuch des PC Herstellers beschrieben.

6 SOFTWARE

Im folgenden Kapitel werden die Software und ihre Verwendung beschrieben.

i

WICHTIG!

Die wichtigsten Informationen für das **Installieren und Deinstallieren der verschiedenen Treiber** finden Sie im mitgelieferten Handbuch "**Installationshinweise für den PCI- und ISA-Bus**".

Sie finden einen Link zu der entsprechenden PDF Datei im Navigationsfenster (Lesezeichen) von Adobe Reader.

Die Karte wird mit einer Treiber-CD-ROM (CD 1) geliefert, die u. a.

- den Treiber und die Software-Beispiele für Windows NT 4.0 und Windows 7/Vista/XP/2000/98 enthält.
- das ADDIREG Registrierungsprogramm für Windows NT 4.0 und Windows 7/Vista/XP/2000/98 enthält.

6.1 Kartenkonfiguration mit ADDIREG

6.1.1 Installationsbeschreibung

Das Registrierungsprogramm ADDIREG ist ein 32-Bit-Programm für Windows 7/Vista/XP/2000/NT4.0/9x. Mit diesem Programm kann der Benutzer alle Hardware-Informationen registrieren, die für die Benutzung der ADDI-DATA PC-Karten erforderlich sind.

i

WICHTIG!

Wenn Sie eine oder mehrere Ressourcen der Karte benutzen, können Sie das ADDIREG Programm nicht starten.

Abb. 6-1: ADDIREG-Hauptfenster (Beispiel)

ADDI-DATA GmbH registration program. Version 0302 / 0546

Resource file System info About

Board list configuration

Board name	Base address	Access	PCI bus/device/[slot]	Interrupt	DMA	More information
APCI1516	D480,DC78,DC40	32-bit	2/9/3	Not available	Not available	ADDIDriver board
APCI1710	D800,DC70	32-bit	2/8/2	17	Not available	

Insert Edit Clear

Board configuration

Base address name : Interrupt name: DMA name: Set Cancel

Base address : Interrupt : DMA channel : Default More information

Access mode: ADDIDriver board manager

Save Restore Test registration Deinstall registration Print registration Quit

ADDI-DATA

Die mittlere Tabelle listet die registrierten Karten und deren Parameter.

Board name:

Die Namen der verschiedenen registrierten Karten werden gezeigt.
(z.B. APCI-1710). Wenn Sie das Programm das erste Mal benutzen, wird keine Karte unter diesem Eintrag aufgelistet.

Base address:

Ausgewählte Basisadresse der Karte.

Access:

Auswahl des Zugriff-Modus für die ADDI-DATA digitalen Karten.
Zugriff in 8-Bit oder 16-Bit Mode.

PCI bus/slot:

Benutzter PCI Steckplatz. Falls die Karte keine APCI-Karte ist, erscheint die Meldung: "NO".

Interrupt:

Benutzter Interrupt der Karte. Falls die Karte keinen Interrupt benutzt, erscheint die folgende Meldung: "Not available".

ISA DMA (nur für ISA-Karten):

Zeigt den ausgewählten DMA-Kanal oder "Not available" an, wenn die Karte keinen benutzt oder wenn die Karte keine ISA Karte ist.

More information:

Weitere Information gibt Ihnen das Dialogfenster, z.B. die Zeichenkette für den Identifier oder die eingebauten COM Schnittstellen.

Textfelder:

Unter der Tabelle befinden sich 6 Text-Eintragfenster, mit denen Sie die Kartenparameter ändern können.

Base address name:

Bezeichnung der benutzten Basisadressen für die Karte. Mit Hilfe des Pulldown-Menüs können Sie den Name auswählen. Der entsprechende Adressbereich wird im unteren Feld angezeigt (Base Address).

Base address:

In diesem Fenster können Sie die Basisadressen Ihrer PC-Karte auswählen. Die freien Basisadressen werden alle aufgelistet. Eine bereits benutzte Basisadresse erscheint nicht unter diesem Eintrag.

Interrupt name:

Wenn die Karte verschiedene Interruptleitungen (Sammel- oder Einzelinterruptleitungen) unterstützen soll, können Sie diese hier auswählen.

Interrupt:

Auswahl der Interruptnummer, welche die Karte benutzen soll.

DMA name (nur für ISA Karten):

Wenn die Karte 2 DMA Kanäle unterstützt, können Sie auswählen, welchen DMA-Kanal Sie ändern.

DMA channel (nur für ISA Karten):

Auswahl des benutzten DMA-Kanals.

Schaltflächen:**Edit¹:**

Auswahl der markierten Karte mit den verschiedenen gesetzten Parametern der Text-Eintragsfenster.

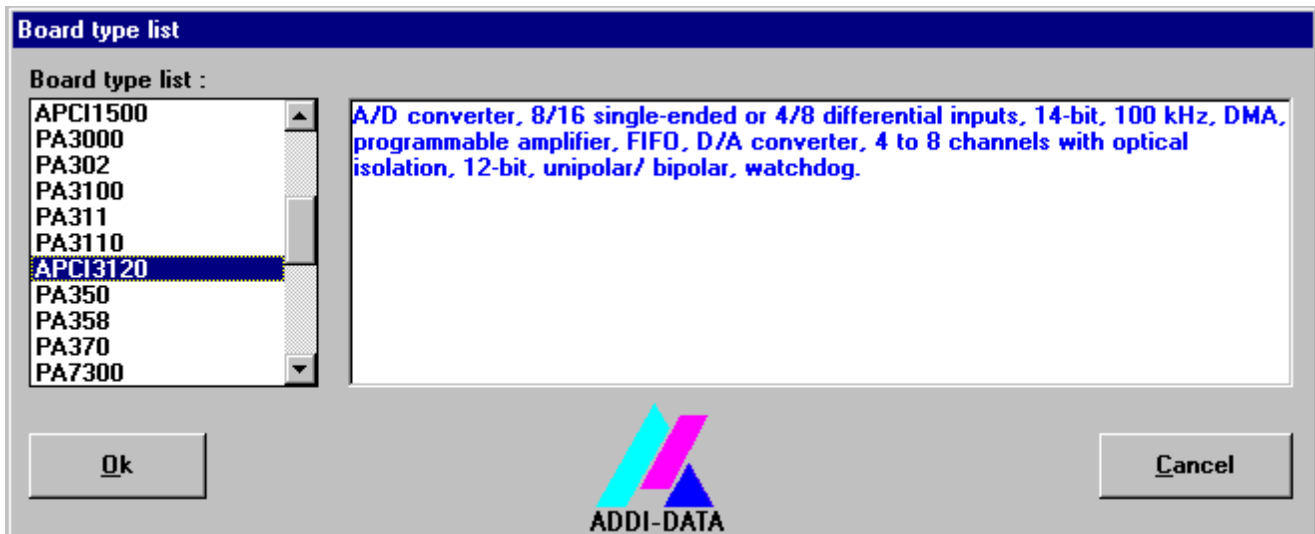
Auf Edit klicken, um die Einträge zu bestätigen oder Doppelklick auf die ausgewählte Karte.

Insert:

Wenn Sie eine neue Karte einfügen wollen, klicken Sie auf "Insert". Das folgende Fenster erscheint am Bildschirm:

¹ "x": Tastenkombination; z.B. "Alt + e" für Edit

Abb. 6-2: Eine neue Karte einbauen



Auf der linken Seite werden alle Karten aufgelistet, die Sie registrieren können. Die ausgewählte Karte anklicken. (Die entsprechende Zeile wird markiert.) Auf der rechten Seite dieses Fensters stehen einige technische Informationen über die Karte zur Verfügung. Bestätigen mit "OK"; Sie kommen zu dem ersten Bildschirm zurück.

Clear:

Sie können die Registrierung der Karte löschen. Markieren Sie die Karte, die Sie löschen wollen und klicken Sie auf "Clear".

Set:

Setzt die parametrierte Kartenkonfiguration. Die Konfiguration soll gesetzt werden, bevor Sie diese speichern.

Cancel:

Setzt die geänderten Parameter auf die momentan gespeicherte Konfiguration zurück.

Default:

Setzt den Standardparameter der Karte.

ADDIDriver Board Manager (nur für die Karten mit ADDIPACK):

Unter Edit/ADDIDriver Board Manager können Sie die aktuellen Einstellungen jeder über den ADDEVICE Manager verwalteten Karten ansehen bzw. modifizieren.

Der ADDevice Manager wird geöffnet. Das Fenster listet alle verfügbaren Ressourcen der **virtuellen Karte** auf.

Save:

Speichert die Parameter und registriert die Karte.

Restore:

Wiederaktivierung der zuletzt gespeicherten Parameter und Registrierung.

Test registration:

Überprüft, ob es einen Konflikt zwischen der Karte und den anderen Geräten gibt. Eine Meldung zeigt den Parameter an, der den Konflikt generiert hat. Wenn es keinen Konflikt gibt, erscheint "OK".

Deinstall registration:

Deinstalliert alle Registrierungen aller Karten aus der Tabelle.

Print registration:

Druckt die Registrierungsparameter auf Ihren Standarddrucker aus.

Quit:

Verlässt das ADDIREG Programm.

More information**i****WICHTIG!**

Je nach Kartentyp stehen dem Anwender mehrere Möglichkeiten zur Verfügung (Siehe nächsten Absatz).

6.1.2 Die Schaltfläche "More information"

Sie können damit kartenspezifische Parameter ändern, z.B. die Identifier Zeichenkette, die COM-Nummer, den Betriebsmode einer Kommunikationskarte, usw.

Wenn Ihre Karte diese Informationen nicht unterstützt, wird diese Schaltfläche gesperrt.

6.1.3 PCI analoge Eingabekarten mit DMA

Wenn Sie die **APCI-3600** im Einsatz haben, wird beim Klicken auf "More information" das unten abgebildete Fenster angezeigt.

Die folgende Abbildung ist ein Beispiel mit 100.000 PCI DMA Erfassungen (im Continuous-Mode).

Für die PCI DMA-Erfassung von analogen Eingängen wird ein linearer Speicher-Buffer des PCs verwendet. Die Buffer-Größe hängt von der Anzahl der Erfassungen ab. Für 1 Erfassung werden 4 Bytes benötigt. Sie können die maximale Anzahl von Erfassungen für Ihre Applikation festlegen und einen großen Buffer zuweisen, nachdem der PC neu gestartet hat.

Für eine Einzelerfassung wird der Karte ein Buffer zugewiesen;
Für eine Continuous Erfassung werden der Karte 2 Buffer zugewiesen.

Abb. 6-3: Verwaltung der PCI DMA-Karten

System informations

Total real memory	:	267952128
Free memory	:	190728856
Number of selected acquisitions	:	100000
Real memory used for PCI DMA	:	400000

PCI DMA board list

Board name bus/device/(slot)	Number of acquisitions	Acquisition mode	DMA buffer size (bytes)	Status
APCI3600 0/14/3	100000	Single	400000	Allocation OK

Single PCI DMA board configuration

Board name : APCI3600 0/14/3

Number of available acquisitions : 47682214

Number of selected acquisitions : Acquisition mode :

System information

Total real memory:

Realer Gesamtspeicher des PCs (in Bytes).

Free memory:

Gibt die Größe (in Bytes) des freiverfügbaren PC-Speichers für die PCI DMA Erfassung zurück.

Number of available acquisitions:

Gibt die Anzahl der möglichen Erfassungen **im Single-Mode** zurück.

Number of selected acquisitions:

Gibt die Anzahl der durch den Anwender ausgewählten Erfassungen zurück.

Real memory used for PCI DMA:

Gibt die Speicher-Größe (in Bytes) zurück, die für die PCI DMA Erfassungen verwendet werden kann.

PCI DMA board list

Liste aller PCI Karten, welche die PCI DMA analoge Eingangserfassung unterstützen. Für jede Karte kann der Benutzer die Anzahl der Erfassungen und den Erfassungsmodus auswählen (Single/Continuous)

Board name:

Kartennamen, Bus-, Device- und Steckplatz-Nummer.

Number of acquisitions:

Anzahl der ausgewählten Erfassungen.

Acquisition mode:

Ausgewählter Erfassungs-Modus (Single oder Continuous).

DMA buffer size (in bytes):

Größe des für diese Konfiguration verwendeten Buffers.

Status:

Not used: Die Anzahl der ausgewählten Erfassungen ist 0

Wait PC restart: Warten darauf, bis der PC neu startet, um Speicher zuzuweisen

Allocation OK: Buffer-Zuweisung OK

Allocation error: Fehler bei der Buffer-Zuweisung. Der Treiber kann der Erfassung keinen linearen Speicher-Buffer zuweisen.

Schaltflächen

Edit:

Auswahl der markierten Karte mit den verschiedenen in "Single PCI DMA board configuration" gesetzten Parametern.

Save:

Speichert die Konfiguration aller Karten.

Quit:

Schließt das Fenster.

Single PCI DMA board configuration: Nachdem Sie eine Karte selektiert haben Edit anklicken:

Single PCI DMA board configuration

Board name:

Kartennamen, Bus-, Device- und Steckplatz-Nummer.

Number of available acquisitions:

Gibt die Anzahl der freien Erfassungen **im ausgewählten Modus** (Erfassungsmodus) **für die nächste Kartenkonfiguration** zurück.

Number of selected acquisitions:

Nummer der selektierten Erfassungen ("Not used" bedeutet, dass kein Buffer für die PCI DMA acquisition zugewiesen wird).

**WICHTIG!**

Je nach Kartentyp stehen dem Anwender mehrere Möglichkeiten zur Verfügung (Siehe nächsten Absatz).

Ungerade Anzahl von Erfassungen wird nicht angenommen und automatisch durch eine annähernde gerade Anzahl ersetzt.

Acquisition mode:

Ausgewählte Erfassungs-Mode:

- Single: Ein einziger Erfassungszyklus wird durchgeführt.
Nach Ablauf des Zyklus wird die Erfassung sofort gestoppt.
- Continuous: Die Erfassung wird durchgeführt, solange die
i_PCI3001_StopAnalogInputAcquisition nicht aufgerufen wird.

**WICHTIG!**

Derzeit ist der „Continuous“-Mode noch nicht verfügbar.

Set:

Setzt die Benutzer-Konfiguration.

Cancel:

Setzt die geänderten Parameter auf die momentan gespeicherte Konfiguration zurück.

6.1.4 Eine neue Karte registrieren



WICHTIG!

Um eine neue Karte zu registrieren, sind Administratorenrechte erforderlich. Nur ein Administrator kann eine neue Karte registrieren oder eine bereits vorhandene Registrierung ändern.

- ◆ Rufen Sie das ADDIREG Programm auf.
- ◆ Klicken Sie auf "Insert".
- ◆ Wählen Sie die gewünschte Karte aus und klicken Sie auf "OK".

Die Default Adresse, Interrupt, und die anderen Parameter werden automatisch gesetzt. Die Parameter werden in den unteren Flächen aufgelistet. Wenn die Parameter nicht automatisch durch das BIOS gesetzt werden, können Sie die Parameter ändern. Klicken Sie dafür auf die gewünschte (-n) Rollfunktion (-en) und wählen Sie einen neuen Wert aus.

Bestätigen Sie mit einem Klick.

- ◆ Wenn die gewünschte Konfiguration gesetzt ist, klicken Sie auf "Set".
- ◆ Die Konfiguration mit "Save" speichern.

Sie können mit einem Test prüfen, ob die Registrierung "OK" ist: Dieses Test überprüft, ob die Registrierung stimmt und ob die Karte vorhanden ist. Wenn der Test erfolgreich war, können Sie das ADDIREG Programm verlassen. Die Karte wird mit den gesetzten Parametern initialisiert und kann betrieben werden.

Müssen in Registrierungsdateien Änderungen vorgenommen werden, so ist es notwendig, den PC neu zu booten. Sie werden durch eine Meldung dazu aufgefordert. Sollte dies nicht nötig sein, können Sie ADDIREG verlassen, und direkt an Ihrer Applikation arbeiten.

6.1.5 Die Registrierung einer vorhandenen Karte ändern



WICHTIG!

Um eine neue Karte zu registrieren, sind Administratorenrechte erforderlich. Nur eine Administrator darf eine neue Karte registrieren oder eine bereits vorhandene Registrierung ändern.

- ◆ Rufen Sie das ADDIREG Programm auf.
- ◆ Markieren Sie die zu ändernde Karte.

Die Parameter der Karte (Basisadresse, DMA Kanal, ..) werden in den unteren Flächen aufgelistet.

- ◆ **Klicken Sie auf die Rollfunktionen der Parameter, die Sie ändern wollen und machen Sie die Roll-Funktionen auf.**
- ◆ **Wählen Sie einen neuen Wert aus.**
- ◆ **Bestätigen Sie mit einem Klick. Wiederholen Sie dies für jeden zu ändernden Parameter.**
- ◆ **Wenn die gewünschte Konfiguration gesetzt ist, klicken Sie auf "Set".**
- ◆ **Die Konfiguration mit "Save" speichern.**

Sie können mit einem Test prüfen, ob die Registrierung "OK" ist: Dieses Test überprüft, ob die Registrierung stimmt und ob die Karte vorhanden ist. Wenn der Test erfolgreich war, können Sie das ADDIREG Programm verlassen. Die Karte wird mit den gesetzten Parametern initialisiert und kann betrieben werden.

Müssen in Registrierungsdateien Änderungen vorgenommen werden, so ist es notwendig, den PC neu zu booten. Sie werden durch eine Meldung dazu aufgefordert.

Sollte dies nicht nötig sein, können Sie ADDIREG verlassen, und direkt an Ihrer Applikation arbeiten.

6.2 Fragen und Software-Download im Internet

Sie können uns Fragen zusenden:

per Email: info@addi-data.com

Download im Internet

Die neueste Version der Standardsoftware für die Karte **APCI-3600** können Sie kostenlos downloaden unter: www.addi-data.com



WICHTIG!

Prüfen Sie vor der Inbetriebnahme und bei evtl. Störungen während des Betriebs, ob für das Produkt ein Update (Handbücher, Treiber) vorliegt. Die aktuellen Daten finden Sie auf unserer Website oder kontaktieren Sie uns direkt.

7 ANSCHLUSS AN DIE PERIPHERIE

7.1 Steckerbelegungen

Abb. 7-1: Koaxialer SMB-Stiftstecker auf Frontblech

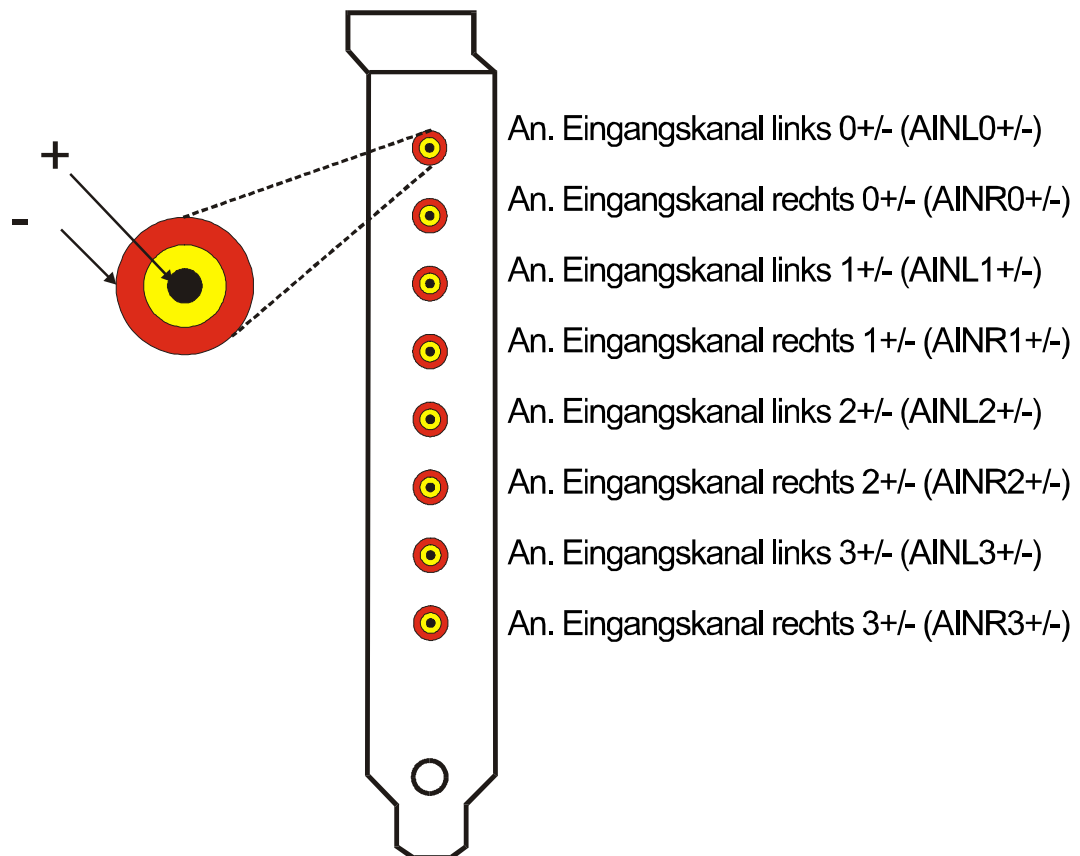


**Abb. 7-2: Digitale Ein- und Ausgänge: 26-pol. Pfostenstecker auf 37-pol. SUB-D-Stiftstecker
(zweites Frontblech: Flachbandkabel FB3600-D)**

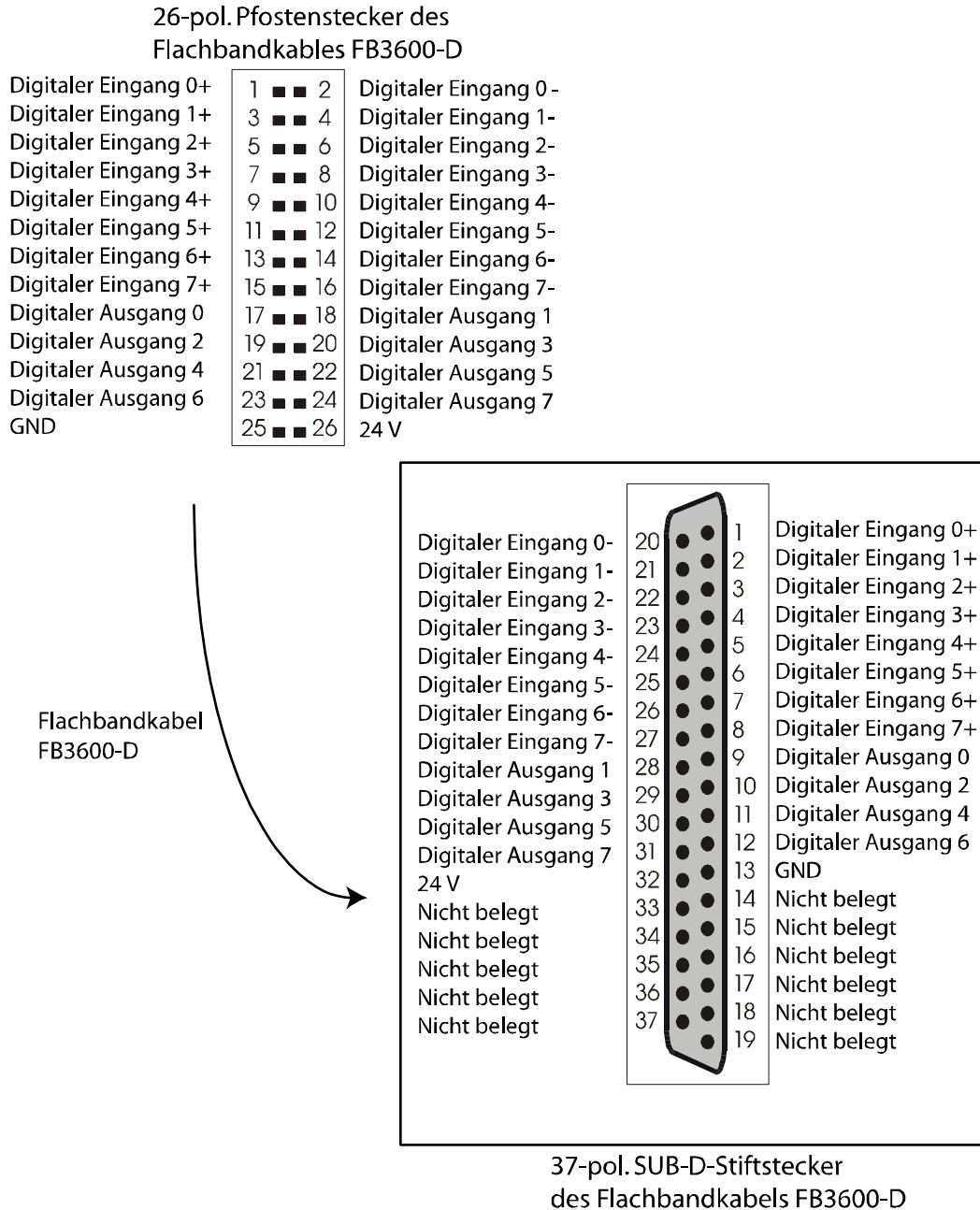


Abb. 7-3: Analoge Ausgänge: 14-pol. Pfostenstecker auf 15-pol. SUB-D Buchsenstecker

Chronometer-Eingänge: 14-pol. Pfostenstecker auf 15-pol. SUB-D Stiftstecker (zweites Frontblech: Flachbandkabel FB3600-AC)

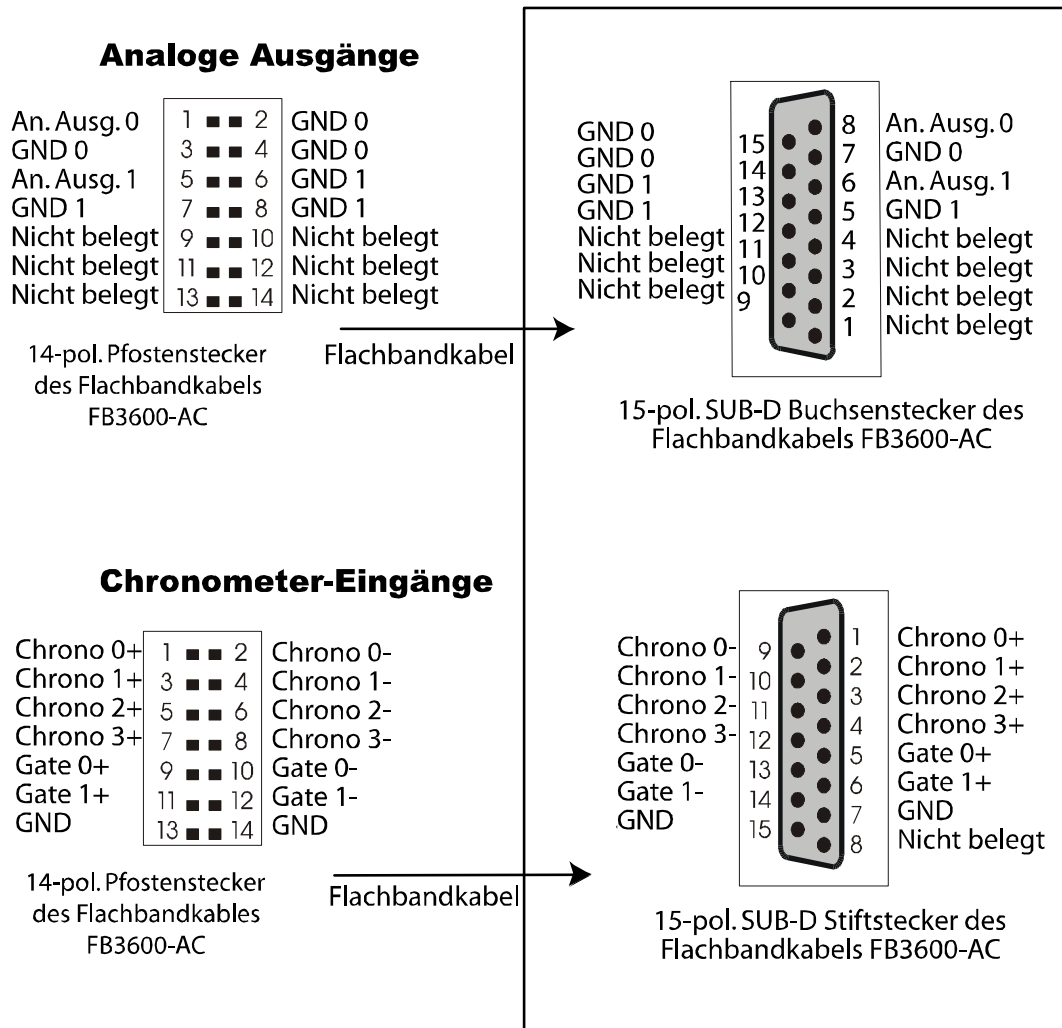
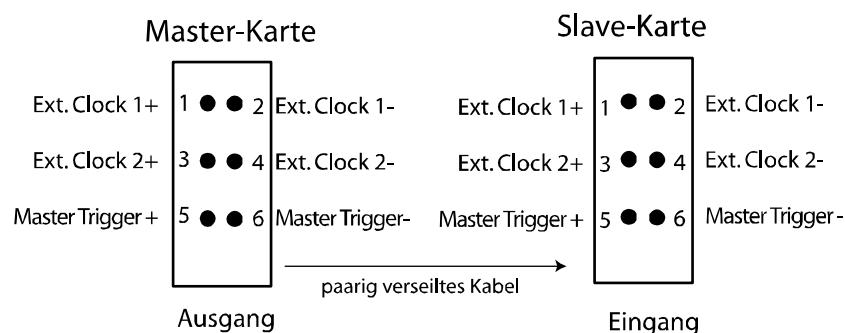


Abb. 7-4: Externe Clock: Pfostenstecker



7.2 Versionsverwaltung

Es gibt zwei Versionen der **APCI-3600** (siehe auch Kapitel 4.3):

- APCI-3600-L:** - 8 analoge Eingänge
- 4 bzw. 8 Stromquellen (ab Revision C, siehe Abb. 4-1) zum Anschluss von ICP-Sensoren
- 128 MBytes SDRAM
- APCI-3600:** - 8 analoge Eingängen
- 4 bzw. 8 Stromquellen (ab Revision C, siehe Abb. 4-1) zum Anschluss von ICP-Sensoren
- 2 analoge Ausgänge, 16-Bit
- 4 Chronometer-Eingänge
- 8 digitale Eingänge, 24 V
- 8 digitale Ausgänge, 24 V
- 128 MBytes SDRAM

8 FUNKTIONEN DER KARTE

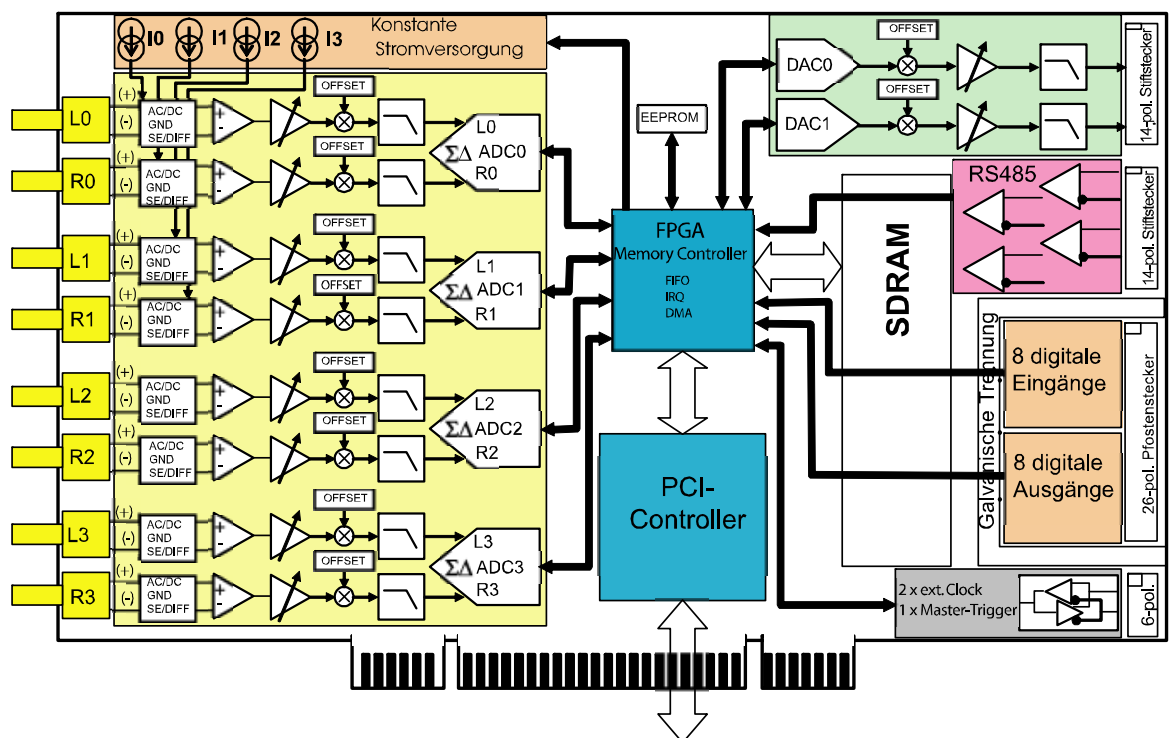
8.1 Überblick

Die APCI-3600 ist eine Multifunktions-PCI-Einsteckkarte mit On Board SDRAM, um analoge Signale im Soundbereich zu erfassen und/oder zu erzeugen. Hierzu stehen Ihnen die folgenden Funktionen zur Verfügung, die in den nachfolgenden Abschnitten näher erläutert werden:

- 8 analoge Eingänge (Kapitel 8.3)
- 2 analoge Ausgänge (Kapitel 8.4)
- 4 Chronometer-Eingänge (Kapitel 8.5)
- 8 digitale Eingänge (Kapitel 8.6)
- 8 digitale Ausgänge (Kapitel 8.7)
- 2 externe Clocks (Kapitel 8.8)
- On Board SDRAM (Kapitel 8.9)

8.2 Blockschaltbild

Abb. 8-1: Blockschaltbild der APCI-3600



8.3 Analoge Eingänge

8.3.1 Verteilung der analogen Eingänge

Die APCI-3600 hat 8 analoge Eingänge, die paarig an 4 Stereo A/D-Wandlern (ADCs) hängen. Zu jedem A/D-Wandler gehört ein linker Kanal (L) und ein rechter Kanal (R).

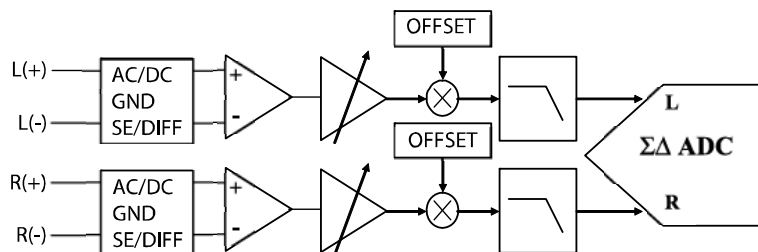
Tabelle 8-1: Kanäle und A/D-Wandler

Kanäle	A/D-Wandler
AINL0, AINR0	ADC0
AINL1, AINR1	ADC1
AINL2, AINR2	ADC2
AINL3, AINR3	ADC3

Jeder Stereo-Eingang besteht aus:

- Einer AC/DC/GND, Single-Ended (SE), differentiellen (DIFF) Schaltung
- Einem differentiellen Eingangsverstärker
- Einer Offset-/Gain-Abgleichsschaltung $\Sigma\Delta$ ADC
- Einem analogen Tiefpassfilter
-

Abb. 8-2: Eingangsschaltung eines analogen Eingangs



8.3.2 Coupling Mode

Jeder Eingang kann im AC-Mode (Alternative Coupling) über einen Hochpassfilter mit einer Eckfrequenz von 0,16 Hz (-3dB) oder im DC-Mode (Direct Coupling) gekoppelt werden.

8.3.3 Konfiguration

Jeder Eingang kann entweder als Single-Ended-Eingang (SE) oder als differentieller Eingang (DIFF) eingestellt werden.

8.3.4 Eingangsbereich

Der Eingangsbereich liegt im folgenden Bereich:

Im SE-Mode: $\pm 10 \text{ V}$

Im DIFF-Mode: $\pm 5 \text{ V}$

8.3.5 Kalibrierung

Die APCI-3600 ermöglicht an allen 8 analogen Eingängen eine Hardware-Kalibrierung der Offset- und Gainfehler (diese Kalibrierung wird von ADDI-DATA durchgeführt).

8.3.6 Anti-Aliasing-Filter

Die APCI-3600 ist mit einem Anti-Aliasing-Filter ausgestattet. Zur Veranschaulichung der Funktionsweise dieser Filterung werden in den folgenden Abschnitten die Basisinformationen zu dieser Filtermethode näher erläutert:

- a) Diskrete Abtastung
- b) Nyquist-Konzept
- c) Anti-Aliasing-Filter
- d) Analogfilter

a) Diskrete Abtastung von Analogsignalen

Das Konzept der diskreten Zeitabtastung und Quantisierung eines analogen Signals besagt Folgendes:

Die kontinuierlichen analogen Daten müssen in diskreten Intervallen abgetastet werden ($t_s = 1/f_s$), die sorgfältig ausgewählt werden sollten, um eine genaue Darstellung des ursprünglichen Analogsignals zu erhalten.

Je mehr Abtastungen vorgenommen werden (schnellere Abtastfrequenzen), desto genauer wird die digitale Darstellung. Wenn jedoch weniger Abtastungen

vorgenommen werden (niedrigere Abtastfrequenzen), wird ein Punkt erreicht, an dem wichtige Informationen über das Signal verloren gehen. Dies führt zum Nyquist-Konzept, das im nachfolgenden Abschnitt beschrieben wird.

b) Nyquist-Konzept

Das Nyquist-Konzept besagt Folgendes:

- Ein analoges Signal mit einer *Bandbreite* f_a muss mit einer Rate von $f_s > 2f_a$ abgetastet werden, da es andernfalls zu Informationsverlusten kommt.
- Der Aliasing-Effekt tritt immer ein, wenn $f_s < 2f_a$ ist.

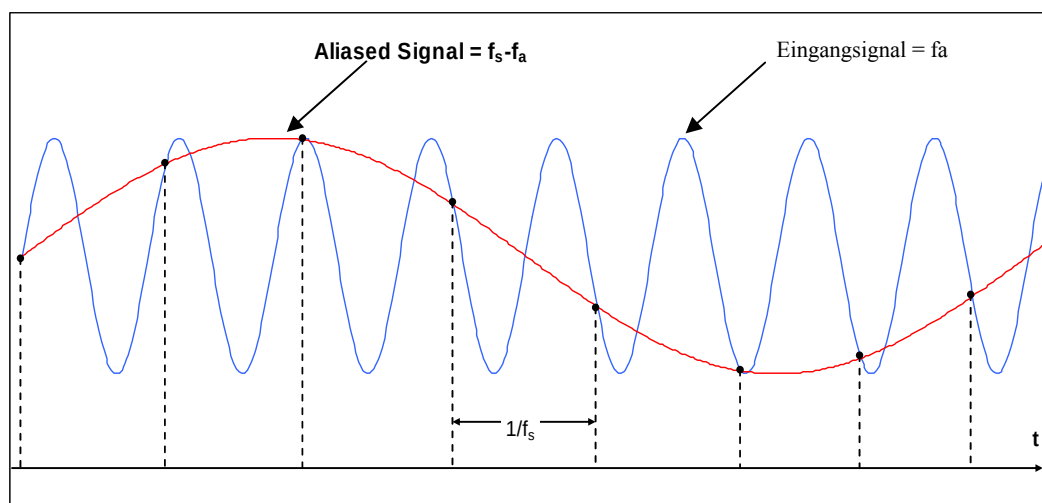
Laut dem Nyquist-Konzept muss die Abtastrate mindestens das Doppelte der Signalbandbreite betragen, um alle Informationen über das Signal zu erhalten, da es andernfalls zu Informationsverlusten kommt. Falls die Abtastrate weniger als das Doppelte der analogen Bandbreite beträgt, tritt der so genannte „Aliasing-Effekt“ ein.

Der Aliasing-Effekt hat sowohl Auswirkungen auf die Zeit als auch auf die Frequenz. Diese beiden Aspekte werden im Folgenden erläutert:

In der Zeitdomain

Die Zeitdomain wird beim Aliasing anhand einer Sinuswelle eines einzelnen aufgenommenen Tons dargestellt (siehe Abb. 8-3). In diesem Beispiel liegt die Abtastrate f_s lediglich leicht über der analogen Eingangsfrequenz f_a und somit wurde das Nyquist-Konzept nicht eingehalten. Es ist zu beachten, dass die hier vorgestellte Struktur der Samples eine Alias-Sinuswelle bei einer niedrigeren Frequenz, entspricht $f_s - f_a$, erzeugt.

Abb. 8-3: Aliasing-Effekt: Zeitlicher Aspekt

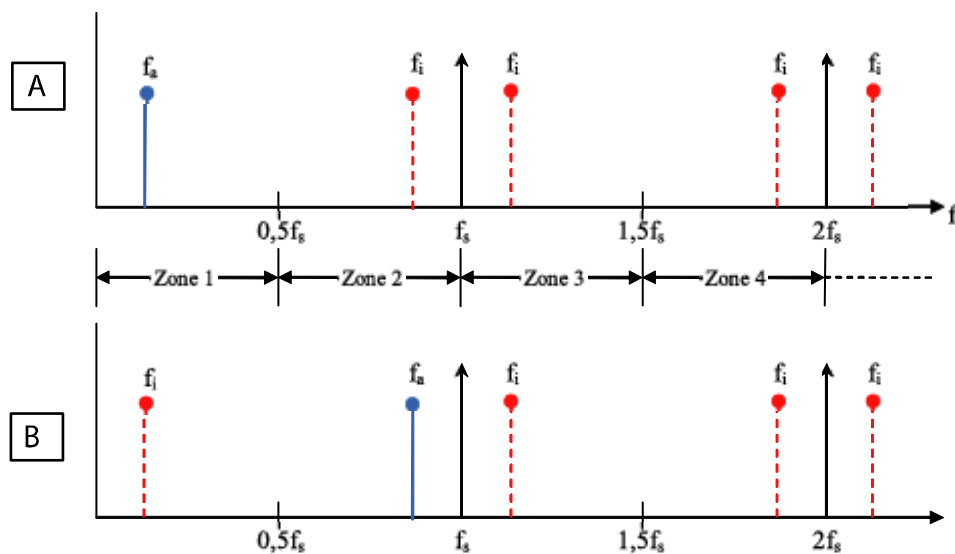


In der Frequenzdomain:

Die Frequenzdomain wird Abb. 8-4/B dargestellt. Hier wird eine einzelne Sinuswellenfrequenz einer Frequenz f_a mit einer Frequenz f_s durch einen Impulssampler abgetastet (siehe Abb. 8-4). Des Weiteren wird angenommen, dass $f_s > 2f_a$ ist. Die Ausgabe der Frequenz des Samplers zeigt Aliasingfrequenzen des Originalsignals um ein Vielfaches von f_s , d.h. bei

Frequenzen gleich $|\pm Kf_s \pm f_a|$, $K = 1, 2, 3, 4$ usw.

Abb. 8-4: Abtastung von Analogsignalen



f_i = Aliasfrequenz (Image)

f_a = Analoge Signalfrequenz

f_s = Samplingfrequenz

Abgetastete Datensysteme:

Die Nyquist-Bandbreite ist das Frequenzspektrum vom DC bis $f_s/2$. Das Frequenzspektrum wird in eine unbegrenzte Anzahl von Nyquist-Zonen eingeteilt, von denen jede eine Breite von $0,5f_s$ hat (siehe Abb. 8-4). D.h., dass der ideale Sampler durch einen A/D-Wandler ersetzt wird, dem ein FFT-Prozessor folgt. Der FFT-Prozessor liefert nur die Ausgabe des DC bis $f_s/2$, d.h. nur die Signale oder Aliasingfrequenzen, die in der ersten Nyquist-Zone aufgetreten sind.

Annahme:

Das Signal liegt außerhalb der ersten Nyquist-Zone (siehe Abb. 8-4/B). In diesem Fall liegt die Signalfrequenz nur leicht unter der Abtastrate, entsprechend den Bedingungen, die in der Darstellung des zeitlichen Aspekts gezeigt werden (siehe Abb. 8-3). Bitte beachten Sie, dass obwohl das Signal außerhalb der ersten Nyquist-Zone liegt, sich seine Aliasingfrequenz (Image) ($f_s - f_a$) innerhalb dieser Zone befindet. Entsprechend der obigen Abbildung (siehe Abb. 8-4/A) ist offensichtlich, dass falls ein unerwünschtes Signal bei einer der dargestellten

Frequenzen von f_a auftaucht, dies auch bei f_a eintreten wird. Dann würde hier in der ersten Nyquist-Zone eine falsche Frequenzkomponente entstehen.

Dies ähnelt dem analogen Mischprozess und bedeutet, dass eine Filterung vor dem Sampler (oder A/D-Wandler) geschehen muss, um Frequenzen, die außerhalb der Nyquist-Bandbreite liegen, aber deren Aliasingfrequenzen innerhalb dieses Bereiches liegen, zu entfernen. Die Filterleistung hängt davon ab, wie nahe das Signal, das außerhalb der Bandbreite auftaucht, bei $f_s/2$ liegt und vom Grad der gewünschten Dämpfung.

c) Anti-Aliasing-Filter

Tiefpassfilter werden entweder vor oder während des Digitalisierungsvorgangs eingesetzt, um alle Frequenzkomponenten, die über der Nyquist-Frequenz liegen, zu entfernen. Dies gewährleistet, dass der digitale Wert oder das digitale Ergebnis frei von allen unerwünschten Frequenzen (Aliasingfrequenzen) ist. Bei der Karte **APCI-3600** werden sowohl digitale als auch analoge Tiefpassfilter eingesetzt, um diese Aliasingfrequenzen zu entfernen.

Die Delta-Sigma-A/D-Wandler auf der **APCI-3600** enthalten integrierte Digitalfilter, deren Eckfrequenz die Abtastrate nachverfolgt. Deshalb passt sich die Filterstruktur automatisch an die Nyquist-Frequenz an. In der nachfolgenden Tabelle werden die Werte für die Eckfrequenz, Passband, Stopband und Stopband-Dämpfung aufgeführt. Der Digitalfilter hat eine nur geringe Auswirkung auf Frequenzen, die innerhalb der Bandbreite liegen. Da der Filter einen FIR (Finite Impulse Response) verwendet, verläuft seine Phasenempfindlichkeit linear.

Tabelle 8-2: Anti-Aliasingfilter: Eingangsstufen

Abtastfrequenz f_s	$2 \text{ kHz} \leq f_s \leq 50 \text{ kHz}$	$50 \text{ kHz} < f_s \leq 100 \text{ kHz}$	$100 \text{ kHz} < f_s \leq 200 \text{ kHz}$
Eckfrequenz (-3 dB)	$0,5 \times f_s$	$0,5 \times f_s$	$0,358 \times f_s$
Passband (-0,1 dB)	DC to $0,47 \times f_s$	DC to $0,45 \times f_s$	DC to $0,24 \times f_s$
Stopband	$0,58 \times f_s$	$0,68 \times f_s$	$0,78 \times f_s$
Stopband-Dämpfung	<95 dB	<92 dB	<97 dB

Abb. 8-5: Single Speed Mode (Stopband-Dämpfung)

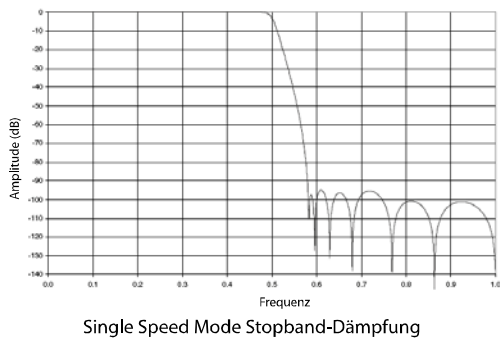


Abb. 8-6: Single Speed Mode (Transition-Band)

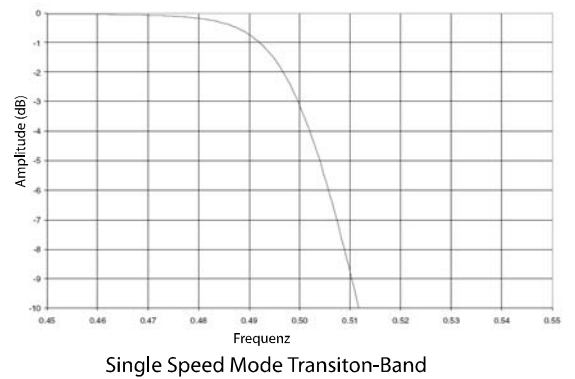


Abb. 8-7: Double Speed Mode (Stopband-Dämpfung)

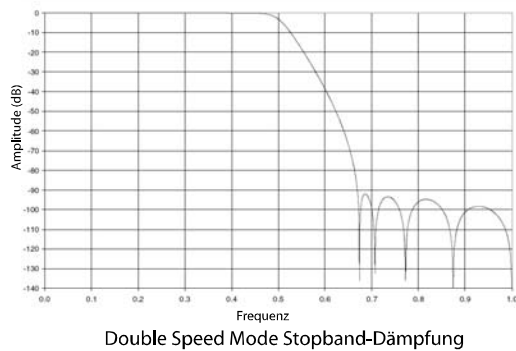


Abb. 8-8: Double Speed Mode (Transition-Band)

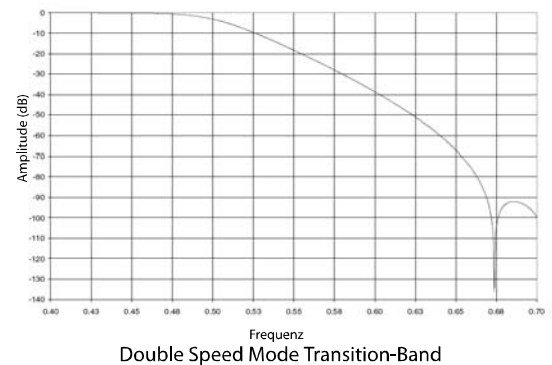


Abb. 8-9: Quad Speed Mode (Stopband-Dämpfung)

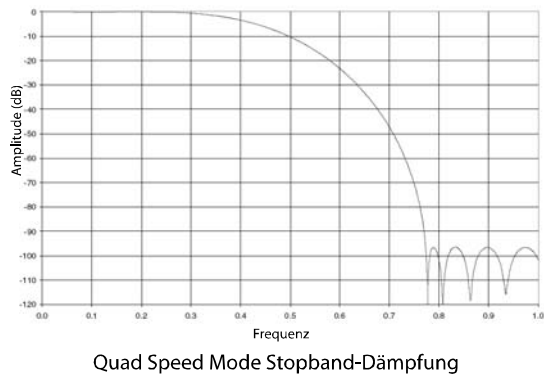
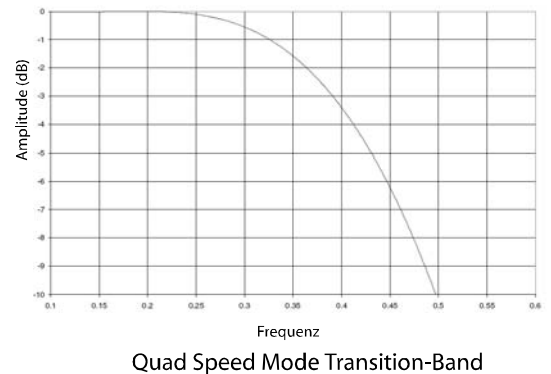


Abb. 8-10: Quad Speed Mode (Transition-Band)



d) Analogfilter

Obwohl der Digitalfilter fast alle Komponenten, die außerhalb der Bandbreite liegen, herausfiltert, ist es dennoch möglich, dass Aliasingfrequenzen von schmalen Frequenzbändern durchkommen. Vor allem von solchen Bändern, die im Bereich einer Nyquist-Bandbreite von $64 \times f_s$ liegen.

Beispiel:

Wenn $f_s = 10.000 \text{ S/s}$, könnte der Digitalfilter Aliasingfrequenzen von Analogelementen zwischen 635 kHz und 645 kHz durchlassen.

Außerdem weisen die Digitalfilter, die in die A/D-Wandler der **APCI-3600** eingebaut sind, auch einen Analogfilter mit einer feststehenden Frequenz auf. Die Analogfilter dienen zur Entfernung von hohen Frequenzen im Analogsignalbereich bevor sie den A/D-Wandler erreichen. Dies eignet sich vor allem für hohe Aliasingfrequenzen von schmalen Bändern, die nicht durch den Digitalfilter abgedeckt werden. Jeder Eingangskanal auf der **APCI-3600** ist mit einem zweipoligen Tiefpassfilter bestückt. Die Eckfrequenz des Analogfilters liegt bei 250 kHz. Dieser Filter hat einen nur geringen Abfall (Roll-Off) mit sehr geringer Dämpfung von jenen Frequenzen, die innerhalb der 45 kHz Eingangsbandbreite der **APCI-3600** liegen. Die hohe Eckfrequenz des Filters gewährleistet einen sehr geringen Amplitudengang und minimale Phasenfehler bei den betroffenen Signalen. Frequenzen, die durch den Digitalfilter gelangt sind, bestehen normalerweise aus Hochfrequenzrauschen. Der analoge Tiefpassfilter entfernt diese hohen Frequenzen bevor sie den A/D-Wandler erreichen.

Während die Eckfrequenz der Digitalfilter direkt mit der Abtastrate mitgeht, ist die Eckfrequenz der Analogfilter fest. Die Eckfrequenz der Analogfilter ist so ausgerichtet, um eine gute Dämpfung von hohen Aliasingfrequenzen zu erreichen bei gleichzeitiger Erhaltung einer geringen Frequenzempfindlichkeit innerhalb des Bandes. Da der Analogfilter ein zweipoliges System ist, ist sein Rolloff nicht sehr scharf. Er weist hervorragende Aliasingdämpfung bei höheren Abtastraten auf, bei denen nur sehr hohe Frequenzen durch den Digitalfilter gekommen sind. Bei geringeren Abtastraten filtert er mögliche Aliasingfrequenzen zwar nicht vollständig heraus, aber meistens bestehen die übrig gebliebenen Aliasingfrequenzen aus Rauschen und nicht aus klaren Oberwellen.

8.3.7 A/D-Wandler

Der A/D-Wandler verwendet einen Multi-Bit Sigma-Delta-Modulator fünfter Ordnung, dem ein Digitalfilter und ein Decimator (Anti-Aliasing-Filter) nachgeschaltet sind.

Auflösung:

Der Sigma-Delta A/D-Wandler hat eine Auflösung von 24-Bit.

Abtastfrequenz:

Die Abtastfrequenz kann zwischen 2 kHz und 200 kHz programmiert werden. Der Sigma-Delta A/D-Wandler besteht aus drei Abtast-Frequenzbereichen (siehe Tabelle 8-3)

Tabelle 8-3: Abtast-Frequenzbereich

Mode	Frequenzbereich
Single Speed	Von 2 kHz bis 50 kHz
Double Speed	Von 50 kHz bis 100 kHz
Quad Speed	Von 100 kHz bis 200 kHz

In allen drei Speed Modes arbeitet der A/D-Wandler mit einer Überabtastungsfrequenz von x 64.

Die Abtastfrequenz ist vom Speed Mode und vom Clock-Divisor-Faktor abhängig. Der Divisor ist für alle Chronometer auswählbar.

Tabelle 8-4 Abtastfrequenz der analogen Eingänge

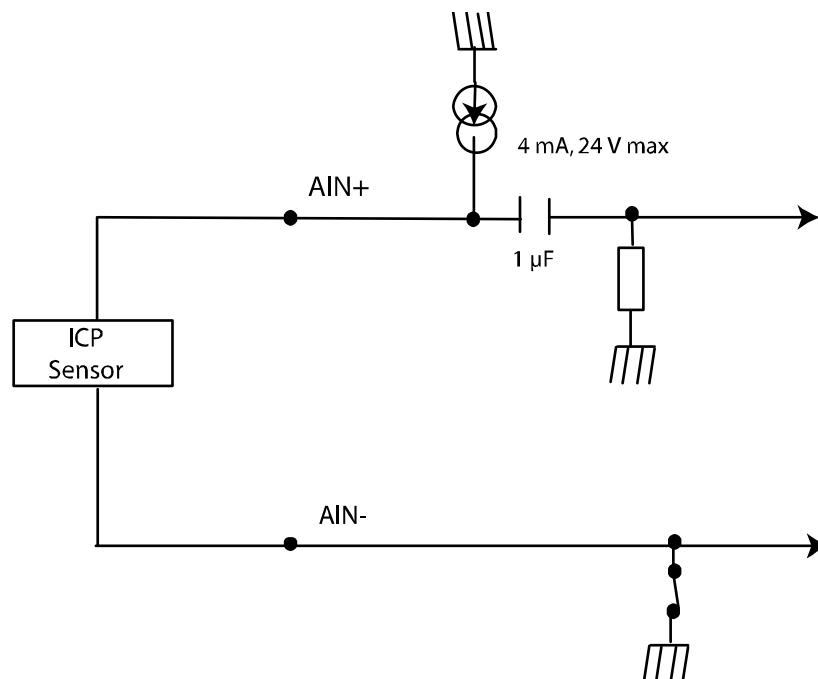
PLD Clock (MHz)	Divisor-Faktor	Clock-Generator (Hz)	Abtastfrequenz f_s (Hz)		
			Single Speed Mode	Double Speed Mode	Quad Speed Mode
102,4	4	25600000	50000	100000	200000
	5	20480000	40000	80000	160000
	6	17066667	33333	66667	133333
	8	12800000	25000	50000	100000
	10	10240000	20000		
	12	8533333	16667		
	16	6400000	12500		
	20	5120000	10000		
	25	4096000	8000		
	40	2560000	5000		
	50	2048000	4000		
	60	1706667	3333		
	80	1280000	2500		
	100	1024000	2000		

8.3.8 Stromquellen

Auf der **APCI-3600** stehen vier bzw. acht Stromquellen (ab Revision C, siehe Abb. 4-1) für die Versorgung der ICP-Sensoren (Integrated Circuit Piezoelectric) zur Verfügung. Die Stromquellen liegen auf den vier ersten positiven analogen Eingängen:

- AINL0+
- AINR0+
- AINL1+
- AINR1+

Die Stromquellen bleiben im DC-Mode automatisch ausgeschaltet und können nur im AC-Mode eingeschaltet werden. Wenn eine Stromquelle aktiviert wird, dann geht der zugehörige positive Eingang in den AC-Mode und der negative Eingang schaltet auf Masse, so dass sich der Stromkreis über den iCP-Sensor bilden kann.

Abb. 8-11: ICP-Sensor Versorgung mit der APCI-3600

8.3.9 Modes

Für die analogen Eingänge der **APCI-3600** stehen Ihnen folgende Modes zur Verfügung:

- a) Simple Mode
- b) Zyklischer Mode: Auto Buffer
- c) Zyklischer Mode: Ring Buffer

a) Simple Mode

In diesem Mode können zwei Werte vom ausgewählten analogen Eingang gelesen werden (linker und rechter Kanal). Der Interrupt wird hier nicht verwendet.

Verwendete Funktionen:

```
i_PCI3600_StartAnalogInputModuleSingleAcquisition
i_PCI3600_GetAnalogInputModuleSingleAcquisitionStatus
i_PCI3600_ReadAnalogInputModuleSingleAcquisitionValues
oder
i_PCI3600_GetAnalogInputModuleValues
```

b) Zyklischer Mode: Auto Stop

Im Auto Stop-Mode (Einzelerfassung) wird die Erfassung gestoppt sobald, die Menge, die Sie vorher für das On Board SDRAM definiert haben, erreicht ist.

Sie können einen Compare-Value mit Hilfe der Software definieren:

Der Compare-Value definiert die Anzahl von Werten für das SDRAM. Wenn diese Werte erreicht sind, wird ein Compare-Interrupt aufgerufen und der Treiber der Karte initialisiert den DMA-Transfer. Sobald der Transfer abgeschlossen ist, wird die User Interrupt Routine aufgerufen. Sobald das On Board SDRAM voll ist, wird ein Auto Stop-Interrupt gesetzt.

Die folgenden Fehler können in diesem Mode auftreten:

- FIFO Error
- PC Buffer Overflow (Wenn der PC-Speicher kleiner als der On Board SDRAM-Speicher ist und die Anwendung des Nutzers den Wert nicht schnell genug gelesen hat)

c) Zyklischer Mode: Ring Buffer

Im Ring Buffer Mode (siehe Kapitel 8.10.2) (kontinuierliche Erfassung), wird die Erfassung nur durch einen Kartenfehler (wie z.B. FIFO-Fehler oder SDRAM-Overflow) oder durch die Software gestoppt.

Sie können einen Compare-Value mit Hilfe der Software definieren:
Der Compare-Value definiert die Anzahl von Werten für das SDRAM. Wenn diese Werte erreicht sind, wird ein Compare-Interrupt aufgerufen und der Treiber der Karte initialisiert den DMA-Transfer des Wertes vom SDRAM zum PC-Speicher. Sobald der Transfer abgeschlossen ist, wird die User Interrupt Routine aufgerufen.

Die folgenden Fehler können in diesem Mode auftreten:

- FIFO Error
- SDRAM Overflow (Wenn der Treiber die Daten nicht schnell genug in den PC-Speicher überträgt).
- PC Buffer Overflow (Wenn der PC-Speicher kleiner als der On Board SDRAM-Speicher ist und die Anwendung des Nutzers den Wert nicht schnell genug gelesen hat)

8.4 Analoge Ausgänge

Die Karte **APCI-3600** verfügt über 2 voneinander unabhängige analoge Ausgänge.

Auflösung:

Jeder DAC hat eine Auflösung von 16-Bit mit einer 13-Bit Genauigkeit.

Ausgangsbereich:

Die Ausgangsspannung liegt im Bereich ± 10 V. Jeder Ausgang kann bis ± 10 mA liefern.

Kalibrierung:

An jedem analogen Ausgang wird der Offset- und der Gain-Fehler über eine Kalibrierung abgeglichen. Die Kalibrierung wird von Addi-Data durchgeführt.

Power-on:

Nach der Power-on-Phase werden die analogen Ausgänge auf 0 V gesetzt.

Abtastfrequenz:

Die Abtastfrequenz ist zwischen 2 kHz und 200 kHz programmierbar. Der DAC hat die folgenden 3 Frequenzbereiche:

Tabelle 8-5: Abtastfrequenz

Mode	Frequenzbereich
Single Speed	Von 2 kHz bis 50 kHz
Double Speed	Von 50 kHz bis 100 kHz
Quad Speed	Von 100 kHz bis 200 kHz

Die Abtastfrequenz ist vom Speed Mode und vom Clock Divisor Faktor abhängig (siehe Tabelle 8-6: Abtastfrequenz der analogen Ausgänge).

8.4.1 Modes der analogen Ausgänge

Für die analogen Ausgänge stehen drei verschiedene Modes zur Verfügung:

- a) Simple Mode
- b) Signalgeneratormode (Free Run)
- c) Signalgeneratormode (Ring Buffer)

a) Simple Mode

Hier werden entweder die Funktionen

i_PCI3600_GetAnalogOutputReadyBitStatus
i_PCI3600_WriteAnalogOutputValue

verwendet oder

i_PCI3600_Set1AnalogOutputChannel

b) Signalgeneratormode (Free Run)

Beim Free Run, wird das definierte SDRAM mit Werten aus dem PC-Speicher gefüllt. Der Wert des SDRAMs wird dann automatisch auf den analogen Ausgang geschrieben. In diesem Fall können keine Fehler auftreten.

c) Signalgeneratormode (Ring Buffer)

Am Anfang (durch die Initialisierung), wird das SDRAM mit Wert aus PC-Speicher gefüllt.

Der Compare-Value legt die Größe des freien Platzes im SDRAM fest. Wenn dieser Wert erreicht wird, wird ein Compare-Interrupt gesetzt und der Treiber der

Karte initialisiert einen DMA-Transfer der Werte vom PC-Speicher zum SDRAM. Sobald der Transfer abgeschlossen ist, wird die Benutzer-Interruptroutine aufgerufen.

In diesem Mode können die folgenden Fehler auftreten:

- FIFO-Fehler
- SDRAM-Underflow : Wenn der Treiber neue Werte nicht schnell genug ins SDRAM transferiert
- PC-Speicher-Underflow : Wenn die Benutzeranwendung neue Werte nicht schnell genug in den PC-Speicher schreibt.

Tabelle 8-6: Abtastfrequenz der analogen Ausgänge

PLD Clock (MHz)	Divisor-Faktor	Clock-Generator (Hz)	Abtastfrequenz f_s (Hz)		
			Single Speed Mode	Single Speed Mode	Quad Speed Mode
102,4	4	25600000	50000	100000	200000
	5	20480000	40000	80000	160000
	6	17066667	33333	66667	133333
	8	12800000	25000	50000	100000
	10	10240000	20000		
	12	8533333	16667		
	16	6400000	12500		
	20	5120000	10000		
	25	4096000	8000		
	40	2560000	5000		
	50	2048000	4000		
	60	1706667	3333		
	80	1280000	2500		
	100	1024000	2000		

8.5 Chronometer-Eingänge

Die Karte **APCI-3600** verfügt über 4 Chronometer-Eingänge (32-Bit) mit RS485-Schnittstelle.

Max. Geschwindigkeit:

1 MHz

Auflösung:

Jeder Chronometer hat eine 32-Bit Registertiefe: auf die steigende Flanke des Eingangssignals wird die Zeit in ein 32-Bit Register gelatcht.

Clear:

Jeder Chronometer kann über ein „Clear“ gelöscht werden. Danach zählt er wieder ab 0. Bei Erreichen des Endwertes ($2^{32} = 4 \text{ GB}$), wird wieder ab 0 gezählt.

Gate-Funktion:

Es stehen zwei Gate-Eingänge mit RS485-Schnittstelle zur Verfügung. Damit können Sie die Chronometer-Eingänge sperren bzw. freigeben.

Gate 0 schaltet Chronometer 0

Gate 1 schaltet Chronometer 1

Ein Low-Pegel an Gate 0 bzw. 1 sperrt den Chronometer-Eingang 0 bzw. 1.

Ein High-Pegel an Gate 0 bzw. 1 gibt den Chronometer-Eingang 0 bzw. 1 frei.

Clock-Divisor:

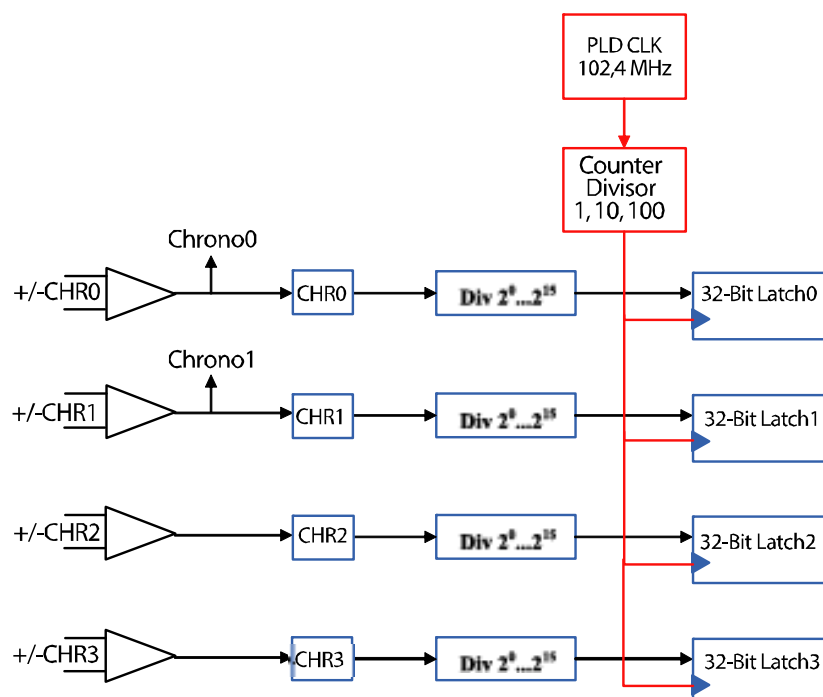
Jeder Chronometer-Eingang wird mit 102,4 MHz getaktet. Dieser Clock kann über Software auch durch 10 oder 100 geteilt werden. Dies wird dann gemeinsam für alle 4 Chronometer eingestellt.

Chronometer-Divisor:

Jeder Chronometer-Eingang hat seinen eigenen Divisor. Dieser kann von 2^0 bis 2^{15} (in 2er Potenz-Schritte) über Software eingestellt werden.

Die beiden ersten Chronometer-Eingänge können auch noch auf den Bus-Clock gelegt werden (siehe Kapitel 8.8)

Abb. 8-12: Chronometer-Eingänge



8.5.1 Modes der Chronometer

Für die Chronometer stehen folgende Modes zur Verfügung:

- a) Simple Mode
- b) Zyklischer Mode: Auto Stop
- c) Zyklischer Mode: Ring Buffer

a) Simple Mode

In diesem Mode werden die Werte im internen FIFO (256 Werte) der Karte gespeichert.

Falls die Software diese Werte nicht liest, tritt ein FIFO-Overflow ein und die Erfassung wird gestoppt. Dieser Mode verwendet nicht das RAM auf der Karte und nicht den Interrupt.

Folgende Softwarefunktionen werden von diesem Mode verwendet:

- i_PCI3600_StartChronometerModuleAcquisition()
- i_PCI3600_GetChronometerModuleFIFOStatus()
- i_PCI3600_ReadChronometerModuleValue()
- i_PCI3600_StopChronometerModuleAcquisition()

b) Zyklischer Mode: Auto Stop

Im Auto Stop-Mode (Einzelerfassung) wird die Erfassung gestoppt sobald, die Menge, die Sie vorher für das On Board SDRAM definiert haben, erreicht ist.

Sie können einen Compare-Value mit Hilfe der Software definieren:

Der Compare-Value definiert die Anzahl von Werten für das SDRAM. Wenn diese Werte erreicht sind, wird ein Compare-Interrupt aufgerufen und der Treiber der Karte initialisiert den DMA-Transfer. Sobald der Transfer abgeschlossen ist, wird die User Interrupt Routine aufgerufen. Sobald das On Board SDRAM voll ist, wird ein Auto Stop-Interrupt gesetzt.

Die folgenden Fehler können in diesem Mode auftreten:

- FIFO Error
- PC Buffer Overflow (Wenn der PC-Speicher kleiner als der On Board SDRAM-Speicher ist und die Anwendung des Nutzers den Wert nicht schnell genug gelesen hat)

c) Zyklischer Mode: Ring Buffer

Im Ring Buffer Mode (kontinuierliche Erfassung), wird die Erfassung nur durch einen Kartenfehler (board Error) (z.b. FIFO Error oder SDRAM-Overflow) oder durch die Software gestoppt.

Sie können einen Compare-Value durch die Software definieren:

Der Compare-Value definiert die Anzahl von Werten für das SDRAM. Wenn diese Werte erreicht sind, wird ein Compare-Interrupt aufgerufen und der Treiber der Karte initialisiert den DMA-Transfer des Wertes vom SDRAM zum PC-

Speicher. Sobald der Transfer abgeschlossen ist, wird die User Interrupt Routine aufgerufen.

Die folgenden Fehler können in diesem Mode auftreten:

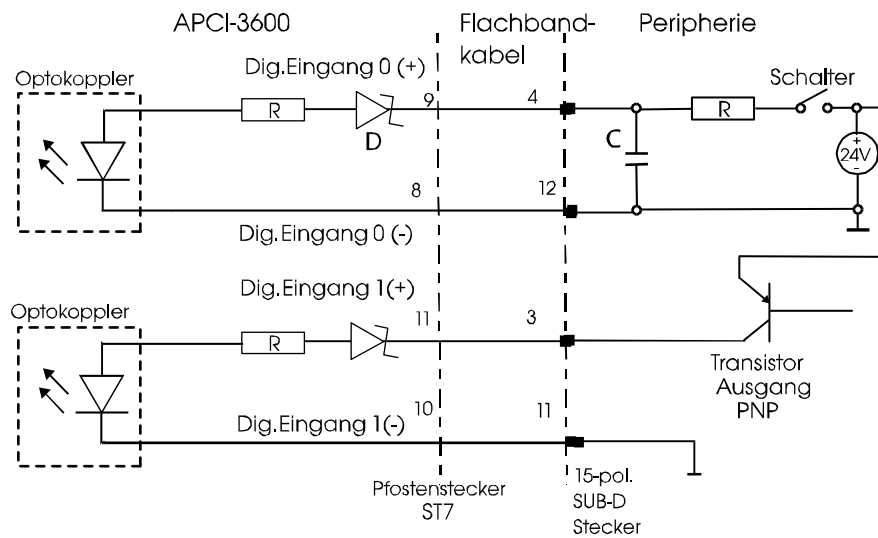
- FIFO Error
- SDRAM Overflow (Wenn der Treiber die Daten nicht schnell genug in den PC-Speicher überträgt).
- PC Buffer Overflow (Wenn der PC-Speicher kleiner als der On Board SDRAM-Speicher ist und die Anwendung des Nutzers den Wert nicht schnell genug gelesen hat)

8.6 Digitale Eingänge

Die Karte **APCI-3600** verfügt über 8 digitale Eingänge

Den ersten digitalen Eingang können Sie als externen Trigger verwenden, um alle Ressourcen der Karte zu starten.

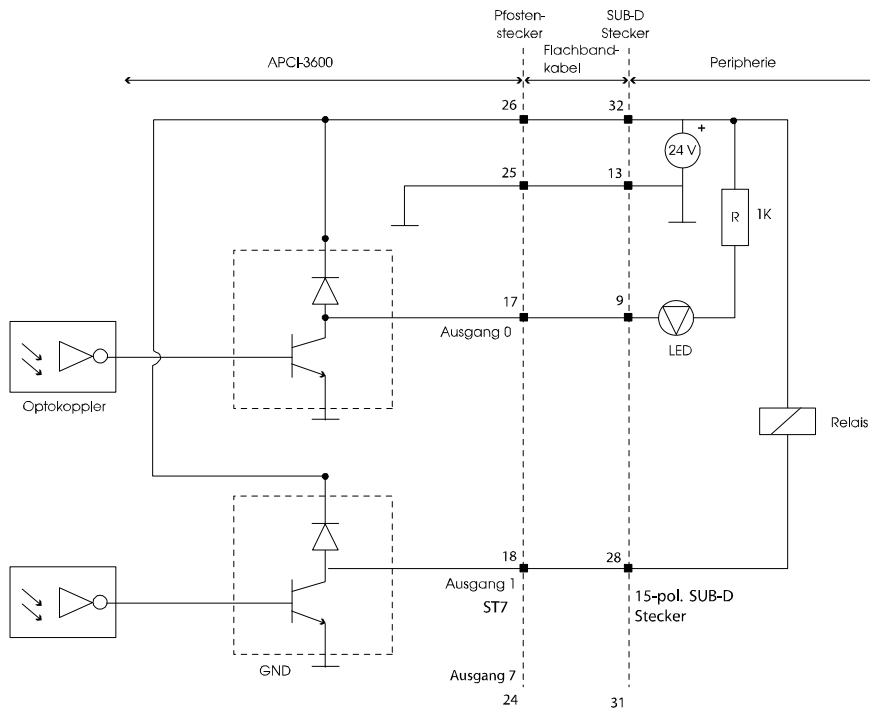
Abb. 8-13: Digitale Eingänge



8.7 Digitale Ausgänge

Die Karte APCI-3600 verfügt über 8 digitale Ausgänge

Abb. 8-14: Digitale Ausgänge



8.8 Externe Clocks und Mastertrigger

Die Karte APCI-3600 verfügt über 2 externe Clocks. Sie können die Karte in den folgenden beiden Modes betreiben:

- Master Mode – Hauptmodus (siehe 8.8.1)
- Slave Mode – Nebenmodus (siehe 8.8.2)

Jeder ADC oder DAC kann einen Bus-Clock aus 2 Bus-Clocks auswählen (Bus-Clocks $\frac{1}{2}$). Jedes Bus-Clock-Signal kann aus 4 voneinander unabhängigen Quellen stammen:

a) aus dem Clock-Generator 1

Stammt vom 102,4 MHz Clock und wird über den Divisor-Generator 1 heruntergetaktet (siehe Tabelle 8-7)

Tabelle 8-7: Clock-Generator Werte)

Tabelle 8-7: Clock-Generator Werte

PLD Clock (MHz)	Divisor Generator 1/2	Clock Generator 1/2 (Hz)
102,4	4	25600000
	5	20480000
	6	17066667
	8	12800000
	10	10240000
	12	8533333
	16	6400000
	20	5120000
	25	4096000
	40	2560000
	50	2048000
	60	1706667
	80	1280000
	100	1024000

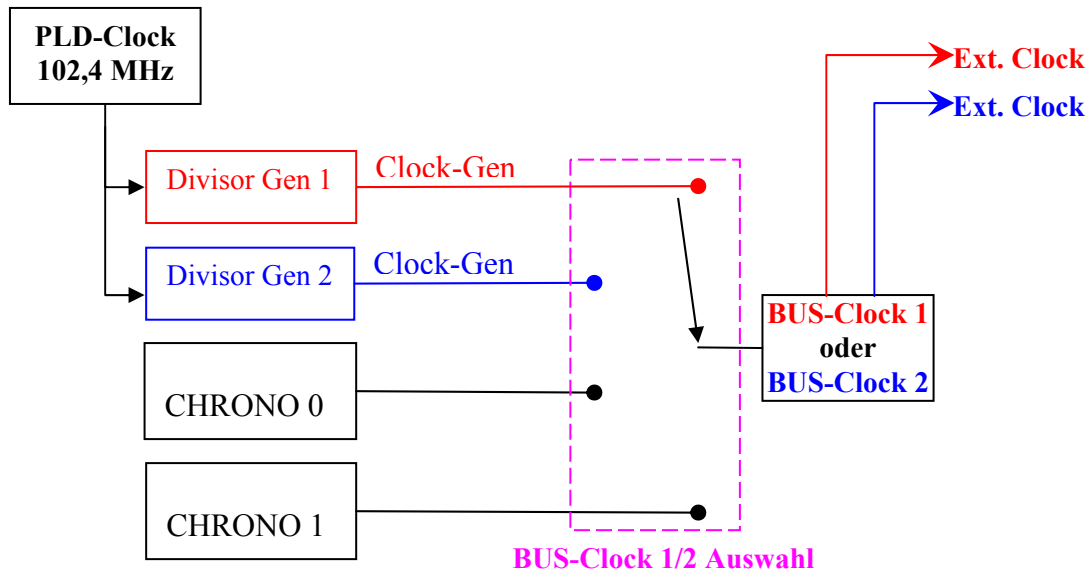
b) aus dem Clock Generator 2

Stammt vom 102,4 MHz Clock und wird über den Divisor Generator 2 heruntergetaktet

c) aus dem Chronometer-Eingang 0**d) aus dem Chronometer-Eingang 1****8.8.1 Master Mode**

Im Master Mode legt die **APCI-3600** den Bus-Clock $\frac{1}{2}$ auf die externen Clocks $\frac{1}{2}$.. Die externen Clocks $\frac{1}{2}$ werden als Ausgang konfiguriert. Wenn Sie die Karte als Master konfigurieren, werden alle Funktionalitäten der Karte im selben Clock synchronisiert: **102,4 MHz.**

Abb. 8-15: Karte im Master Mode

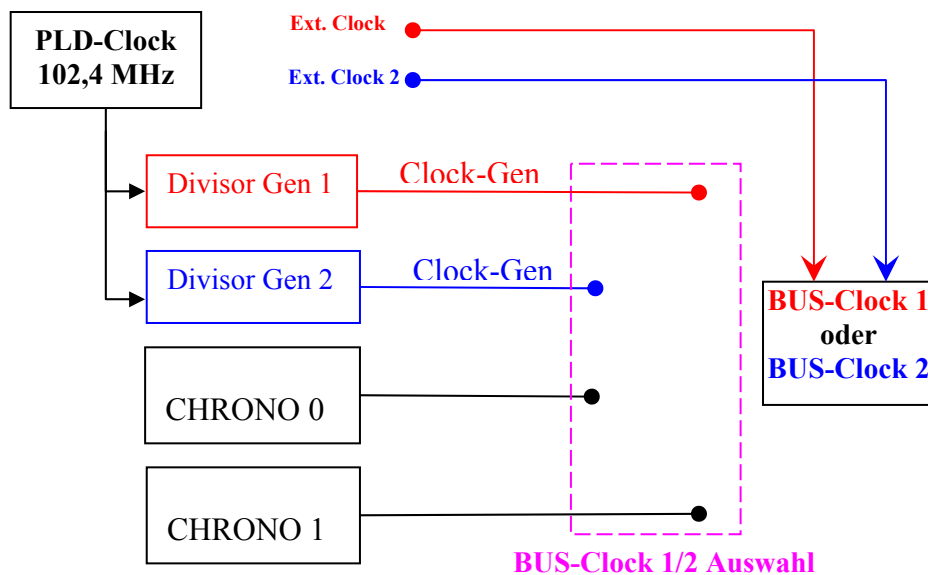


8.8.2 Slave Mode

Im Slave-Mode werden die Bus-Clocks $\frac{1}{2}$ direkt an die externen Clocks $\frac{1}{2}$ angeschlossen. Die externen Clocks $\frac{1}{2}$ werden als Eingang konfiguriert. In diesem Mode können mehrere **APCI-3600** synchronisiert werden. Dazu wird eine Karte im Master Mode eingestellt und alle anderen im Slave Mode.

Wenn Sie die Karte als Slave konfigurieren, werden die analogen Eingangs- und Ausgangskanäle mit demselben Clock der Masterkarte konfiguriert. Der Chronometer verwendet immer den Clock auf der Slave-Karte: **102,4 MHz**.

Abb. 8-16: Karte im Slave Mode



8.9 On Board-Speicher (SDRAM)

Die **APCI-3600** unterstützt SDRAM-Speicher, um den hohen Datenfluss von der Karte zum Rechner zu übertragen.

Der On Board-Speicher ist ein 144 pin SO-DIMM SDRAM-Modul (Notebook-Speicher). Die **APCI-3600** wird mit einem 128 MBytes SDRAM-Modul in der Standardversion ausgeliefert. Die **APCI-3600** unterstützt 144 pin SO-DIMM SDRAM-Modul mit unterschiedlichen Speichertiefen/-größen:

- 64 MBytes
- 128 MBytes (Standard)
- 256 MBytes
- 512 MBytes

8.10 Speicherkonzept

Im Folgenden werden die grundlegenden Speicherkonzepte der **APCI-3600**-Karte vorgestellt:

- Globaler Speicher
- Ring Buffer

8.10.1 Speicherkonzept: Globaler Speicher

Nach dem Einbau Ihrer Karte, müssen Sie diese mit dem ADDIREG-Programm registrieren (siehe Kapitel 6.1)

Falls Sie die zyklische Erfassung oder den Signalgenerator der Karte verwenden möchten, müssen Sie hierfür zuerst Speicherplatz (RAM) auf Ihrem Rechner reservieren.

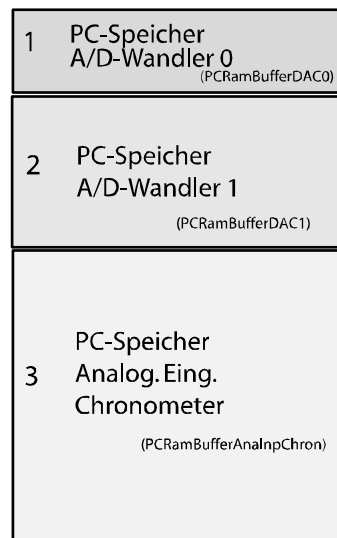
Im ADDIREG-Programm können Sie die Größe des globalen Speichers festlegen, der dem RAM im Rechner zugewiesen wird.

Dieser Speicher wird „PCRamGlobalBuffer“ (PC RAM globaler Speicher) genannt.

Die Größe dieses Speichers wird nach folgenden Kriterien festgelegt:

- Die Speichergröße, die Sie durch jeden Interrupt für jeden A/D-Wandler/Chronometer erhalten möchten
- Die Speichergröße, die für jeden A/D-Wandler reserviert werden soll

Abb. 8-17: Globaler Speicher



Der PC-Speicher „analoge Eingänge/Chronometer“ (3) (PCRamBufferAnaInpChrono Buffer) wird gemeinsam von den analogen Eingängen und dem Chronometer verwendet.

8.10.2 Speicherkonzept: Ring Buffer

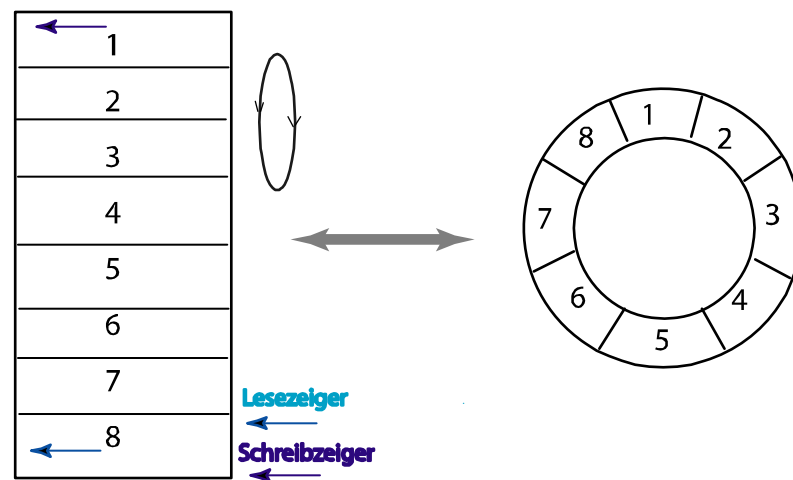
Im Folgenden werden zuerst das Ring Buffer-Konzept im Allgemeinen und danach auf der **APCI-3600** dargestellt.

a) Überblick: Ring Buffer

Ein Ring Buffer ist der Teil eines Speichers mit FIFO-Zugriff (FIFO: First In First Out: Beim Leeren eines Speichers werden zuerst eingespeicherte Daten als erste wieder ausgegeben).

Ein Ring Buffer besteht aus folgenden Komponenten:

- Speicherteil mit einer definierten Größe
- Einem Lesezeiger
- Einem Schreibzeiger
- Flags: Leer und gefüllt

Abb. 8-18: Ring Buffer**Handhabung der Flags:**

Der Leseprozess setzt das leere Flag, wenn nach dem Lesen der Lesezeiger mit dem Schreibzeiger übereinstimmt. Der Leseprozess setzt das gefüllte Flag zurück. Der Schreibprozess setzt das gefüllte Flag, wenn nach dem Schreiben der Schreibzeiger mit dem Lesezeiger übereinstimmt. Der Schreibprozess setzt das leere Flag zurück.

Verwendung des Ring Buffers:

Der Ring Buffer wird verwendet, wenn zwei Prozesse Daten miteinander austauschen möchten und wenn diese beiden Prozesse asynchron laufen.

b) Ring Buffer auf der APCI-3600

Wenn der Ring Buffer Mode für die analogen Eingänge und Ausgänge sowie für den Chronometer verwendet wird, geschieht dies in folgender Weise:
Das On Board SDRAM und der PC-Speicher werden als Ring Buffer verwendet.

On Board SDRAM Ring Buffer:

- Die Firmware verwaltet die Flags des Ring Buffers des On Board SDRAMs
- Die Firmware erzeugt einen Interrupt durch Overflow/Underflow (Überfüllung/Entleerung) des Ring Buffers und stoppt die Funktionalität, die diesen Buffer verwendet.

Die beiden Prozesse sind:

- Die Firmware der Karte
- Der Treiber der Karte (Interrupt-Funktion)

PC-Speicher Ring Buffer:

Der Treiber der Karte (Interrupt-Funktion) startet einen DMA-Transfer, um neue Werte in den PC-Speicher zu schreiben.

Die Benutzeranwendung liest den Wert vom PC-Speicher. Bei einem Overflow des PC-Speichers stoppt der Treiber der Karte diese Ressource und informiert die Benutzeranwendung, indem er die Benutzer-Interruptroutine aufruft. Der Treiber der Karte verwaltet die Flags des PC-Speichers.

Die beiden Prozesse sind:

- Der Treiber der Karte (Interruptfunktion)
- Die Benutzeranwendung

8.11 Maximale Datenübertragungsgeschwindigkeit

Die Übertragungsgeschwindigkeit der **APCI-3600** ist von unterschiedlichen Parametern abhängig, wie z.B.:

- der Abtastfrequenz
- der Anzahl der Kanäle
- dem Erfassungsmode

Im ungünstigsten Fall werden alle Kanäle im Ring Buffer-Mode funktionieren, mit einer maximalen Abtastfrequenz von 200 kHz, d.h.

- 8 analoge Eingänge mit jeweils 32-Bit pro Kanal und 200 kHz Abtastfrequenz
- 2 analoge Ausgänge mit jeweils 32-Bit pro Kanal und 200 kHz Abtastfrequenz
- 4 Chronometer-Eingänge mit jeweils 32-Bit pro Kanal und 1 MHz Eingangsfrequenz

Tabelle 8-8: Datenübertragungsgeschwindigkeit

Funktion	Kanäle	Tiefe	Max. Abtastfrequenz	Datentransfer
Analog. Eingang	8	32-Bit	200 kHz	6,4 MB/s
Analog. Ausgang	2	32-Bit	200 kHz	1,6 MB/s
Chronometer	4	32-Bit	1 MHz	16 MB/s

Im ungünstigsten Fall werden dann **24 MBytes/s** übertragen. Um diese Rate zu erreichen braucht Ihre Rechner folgende Voraussetzungen:

- einen schnellen Prozessor
- eine schnelle Festplatte
- ausreichend SDRAM-Speicher, um einen großen PC-Buffer für die **APCI-3600** zuteilen zu können

9 SOFTWARE

9.1 Softwarefunktionen

9.1.1 Allgemeine Funktionen

1) i_PCI3600_InitCompiler()

Syntax:

INT i_PCI3600_InitCompiler(BYTE b_CompilerDefine)

Parameter:**- Eingabe:**

BYTE b_CompilerDefine Der Benutzer soll die Sprache unter Windows auswählen, in der er programmieren möchte.

- DLL_COMPILER_C:
Der Benutzer programmiert in C.
- DLL_COMPILER_VB_4:
Der Benutzer programmiert in Visual Basic for Windows.
- DLL_COMPILER_VB_5_VB_6:
Der Benutzer programmiert in Visual Basic 5/6 für Windows 7/Vista/NT/2000/XP oder Windows 98.
- DLL_COMPILER_PASCAL:
Der Benutzer programmiert in Pascal oder Delphi.
- DLL_LABVIEW :
Der Benutzer programmiert in Labview.
- DLL_COMPILER_DOT_NET :
Der Benutzer programmiert in .NET

- Ausgabe:

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert den verwendeten Compiler.

Funktionsaufruf:

INT i_ReturnValue;

i_ReturnValue = i_PCI3600_InitCompiler (DLL_COMPILER_C);

Return-Wert:

0 : Kein Fehler

-1 : Compiler-Parameter ist falsch

2) i_PCI3600_GetBoardList ()

Syntax:

```
_INT_ i_PCI3600_GetBoardList (PBYTE pb_NbrOfBoard,  
                               PBYTE pb_PCISlotNbr)
```

Parameter:

- Eingabe:

Es erfolgt keine Eingabe

- Ausgabe

PBYTE pb_NbrOfBoard	Gibt die Anzahl der gefundenen Karten zurück
PBYTE pb_PCISlotNbr	Gibt die PCI-Steckplatznummer jeder gefundenen Karte zurück

Aufgabe:

Gibt die Anzahl der gefundenen **APCI-3600**-Karten (*pb_NbrOfBoard*) und die PCI-Steckplatznummer von jeder Karte zurück (*pb_PCISlotNbr*).

Funktionsaufruf:

```
BYTE b_NbrOfBoard;  
BYTE b_PCISlotArray [20];  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_GetBoardList (&b_NbrOfBoard, b_PCISlotArray);
```

Return-Wert:

0 : Kein Fehler

3) i_PCI3600_OpenBoard ()

Syntax:

```
_INT_ i_PCI3600_OpenBoard (BYTE b_BoardIndex,  
                           PDWORD pdw_BoardHandle)
```

Parameters:

- Eingabe:

BYTE b_BoardIndex Index der zu öffnenden Karte

- Ausgabe

PDWORD pdw_BoardHandle Handle der Karte **APCI-3600** zur Verwendung der Funktionen

Aufgabe:

Öffnet die Karte mit dem Index: *b_BoardIndex*. Ein Handle wird an den Benutzer zurückgegeben, um die folgenden Funktionen zu benutzen. Handles ermöglichen die Verwaltung mehrerer Karten.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT        i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_OpenBoard (0,  
                                     &dw_BoardHandle);
```

Return-Wert:

0 : Kein Fehler

-1: Kartenindex ist nicht verfügbar

-2 : Karte nicht vorhanden

-3: Kein Handle für die Karte verfügbar

-4: Kann den Windows-Treiber nicht öffnen

-5: Zeichnet den physikalischen Speicherfehler auf

-6: Diese Kartenversion wird von diesem Treiber nicht unterstützt. Bitte aktualisieren Sie ihn.

-7 : Fehler bei der Speicherzuweisung für die Descriptor-Liste, die beim DMA-Transfer verwendet wurde.

-8 : Fehler beim Erzeugen End-Debug Rückruf-Funktion

-9 : Fehler beim Installieren der End-Debug Rückruf-Funktion

-10: Kann den Mutex-Treiber nicht erzeugen

-11: Fehler bei der Zuteilung des Gemeinschaftsspeichers

-12 : Fehler bei der Erzeugung der Mutex-Funktion

-13: Allgemeiner Fehler beim Öffnen

4) i_PCI3600_InitBoard 0

Syntax:

```
_INT_ i_PCI3600_InitBoard (DWORD dw_BoardHandle,
                           BYTE b_MasterSlaveMode,
                           BYTE b_ClockGen1_Divisor,
                           BYTE b_ClockGen2_Divisor,
                           BYTE b_SamplingClock1_Selection,
                           BYTE b_SamplingClock2_Selection,
                           BYTE b_SlaveSamplingClock1_Selection,
                           BYTE b_SlaveSamplingClock2_Selection,
                           BYTE b_Chronometer_MasterClockDivisor)
```

Parameter:
- Eingabe:

DWORD	dw_BoardHandle	Handle der Karte
BYTE	b_MasterSlaveMode	Auswahl des Modes der Karte: 0 : Master Mode 1 : Slave Mode Im Slave Mode, muss die Auswahl von Clock1 und 2 und des Divisors dieselbe wie bei der Masterkarte sein.
BYTE	b_ClockGen1_Divisor	Auswahl des Divisors für den Clock-Generator 1. Siehe Tabelle 9-1
BYTE	b_ClockGen2_Divisor	Auswahl des Divisors für den Clock-Generator 2 Siehe Tabelle 9-1
BYTE	b_SamplingClock1_Selection	Auswahl des Clocks für den Sampling-Clock 1 Siehe Tabelle 9-2
BYTE	b_SamplingClock2_Selection	Auswahl des Clocks für den Sampling-Clock 2 Tabelle 9-2
BYTE	b_SlaveSamplingClock1_Selection	Auswahl des Clocks für den Sampling-Clock 1 der Slave-Karte 0 : PCI3600_EXT_CLOCK_1 1 : PCI3600_EXT_CLOCK_2
BYTE	b_SlaveSamplingClock2_Selection	Auswahl des Clocks für den Sampling-Clock 2 der Slave-Karte. 0 : PCI3600_EXT_CLOCK_1 1 : PCI3600_EXT_CLOCK_2
BYTE	b_Chronometer_MasterClockDivisor	Auswahl des Divisors für den Master-Clock des Chronometers: 0 : PCI3600_CHRONOMETER_DIVISOR_1 1 : PCI3600_CHRONOMETER_DIVISOR_10

2 :
PCI3600_CHRONOMETER_DIVISOR_100

- Ausgabe

Es erfolgt keine Ausgabe.

Aufgabe:

Initialisiert die Karte: Initialisierung der beiden Master-Clocks, die von den analogen Ein- und Ausgängen verwendet werden und des Divisors für die Verwendung des Clocks beim Chronometer

Definiert außerdem, ob die Karte als Master- oder Slave-Karte arbeitet.

Tabelle 9-1: Clock-Divisor

Bezeichnung	Erklärung: Clock-Generator, Werte in MHz
PCI3600_DIVISOR_4	25,600
PCI3600_DIVISOR_5	20,480
PCI3600_DIVISOR_6	17,060
PCI3600_DIVISOR_8	12,800
PCI3600_DIVISOR_10	10,240
PCI3600_DIVISOR_12	8,530
PCI3600_DIVISOR_16	6,400
PCI3600_DIVISOR_20	5,120
PCI3600_DIVISOR_25	4,096
PCI3600_DIVISOR_40	2,560
PCI3600_DIVISOR_50	2,048
PCI3600_DIVISOR_60	1,706
PCI3600_DIVISOR_80	1,280
PCI3600_DIVISOR_100	1,024

Tabelle 9-2: Clock-Auswahl

Bezeichnung	Erklärung
PCI3600_CLK_GENERATOR_1	Clock generator 1 (Clock-Generator 1) = 25,6MHz / Clock Gen 1 Divisor
PCI3600_CLK_GENERATOR_2	Clock generator 2 (Clock-Generator 2) = 25,6MHz / Clock Gen 2 Divisor
PCI3600_CHRONOMETER_0_LATCH_INPUT	Chronometer 0 latch input
PCI3600_CHRONOMETER_1_LATCH_INPUT	Chronometer 1 latch input

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

```
i_ReturnValue = i_PCI3600_InitBoard (dw_BoardHandle,  
                                     0,  
                                     PCI3600_DIVISOR_1,  
                                     PCI3600_DIVISOR_10,  
                                     PCI3600_CLK_GENERATOR_1,  
                                     PCI3600_CLK_GENERATOR_2,  
                                     0,  
                                     0,  
  
                                     PCI3600_CHRONOMETER_DIVISOR_1);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch.

-2: Master./Slave-Mode-Parameter ist falsch.

-3: Einer oder beide der ausgewählten Parameter des Bus-Clock-Divisors sind falsch.

-4: Die Auswahl von einem oder beiden Clocks ist falsch.

-5: Auswahl des Slave-Sampling-Clocks ist falsch

-6: Parameter des Chronometer-Master-Clock-Divisors ist falsch

-7: Die Karte wurde schon durch einen anderen Prozess initialisiert.

-8: Die Karte wird verwendet: Setzen Sie alle Kanäle frei, um sie neu initialisieren zu können

5) i_PCI3600_GetBoardInitialisation ()

Syntax:

```

_INT_ i_PCI3600_GetBoardInitialisation (DWORD dw_BoardHandle,
                                         PBYTE pb_MasterSlaveMode,
                                         PBYTE pb_ClockGen1_Divisor,
                                         PBYTE pb_ClockGen2_Divisor,
                                         PBYTE
pb_SamplingClock1_Selection,
                                         PBYTE
pb_SamplingClock2_Selection,
                                         PBYTE_
pb_SlaveSamplingClock1_Selection,
                                         PBYTE_
pb_SlaveSamplingClock2_Selection,
                                         PBYTE_
pb_Chronometer_MasterClockDivisor)

```

Parameter:

- Eingabe:

DWORD dw_BoardHandle Handle der Karte

- Ausgabe

PBYTE	pb_MasterSlaveMode	Modes der Karte : 0 : Master-Mode 1 : Slave-Mode
PBYTE	pb_ClockGen1_Divisor	Divisor des Clock-Generators 1 (Siehe Tabelle 9-1)
PBYTE	pb_ClockGen2_Divisor	Divisor des Clock-Generators 2. (Siehe Tabelle 9-1)
PBYTE	pb_SamplingClock1_Selection	Clock für Sampling-Clock 1 (Siehe Tabelle 9-1 und Tabelle 9-2)
PBYTE	pb_SamplingClock2_Selection	Clock für den sampling clock 2 (Siehe Tabelle 9-2)
PBYTE	pb_SlaveSamplingClock1_Selection	Clock für den Sampling-Clock 1 der Slave-Karte 0 : PCI3600_EXT_CLOCK_1 1 : PCI3600_EXT_CLOCK_2
PBYTE	pb_SlaveSamplingClock2_Selection	Clock für den Sampling-Clock 2 für die Slave-Karte 0 : PCI3600_EXT_CLOCK_1 1 : PCI3600_EXT_CLOCK_2
PBYTE	pb_Chronometer_MasterClockDivisor	

Divisor für den Master-Clock für den
Chronometer:
0 : PCI3600_CHRONOMETER_DIVISOR_1
1 :
PCI3600_CHRONOMETER_DIVISOR_10
2 :
PCI3600_CHRONOMETER_DIVISOR_100

Aufgabe:

Initialisierungsinformationen der Karte.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;  
BYTE b_MasterSlaveMode;  
BYTE b_ClockGen1_Divisor;  
BYTE b_ClockGen2_Divisor;  
BYTE b_SamplingClock1_Selection;  
BYTE b_SamplingClock2_Selection;  
BYTE b_SlaveSamplingClock1_Selection;  
BYTE b_SlaveSamplingClock2_Selection;  
BYTE b_Chronometer_MasterClockDivisor;
```

```
i_ReturnValue = i_PCI3600_GetBoardInitialisation (dw_BoardHandle,  
                                                    &b_MasterSlaveMode,  
                                                    &b_ClockGen1_Divisor,  
                                                    &b_ClockGen2_Divisor,  
  
                                                    &b_SamplingClock1_Selection,  
  
                                                    &b_SamplingClock2_Selection,  
  
                                                    &b_SlaveSamplingClock1_Selection,  
  
                                                    &b_SlaveSamplingClock2_Selection,  
  
                                                    &b_Chronometer_MasterClockDivisor);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch.
-2: Die Karte ist nicht initialisiert

6) i_PCI3600_GetBoardInformation ()**Syntax:**

```

_INT_ i_PCI3600_GetBoardInformation(DWORD dw_BoardHandle,
                                     PDWORD
pdw_OnBoardRAMTotalSize,
                                     PBYTE
pb_NbrOfComputerBufferAllocated,
                                     PDWORD
pdw_ComputerBufferSizeArray )

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte

- Ausgabe:

PDWORD pdw_OnBoardRAMTotalSize
 Gibt die RAM-Größe der Karte zurück (in MBytes).

PBYTE pb_NbrOfComputerBufferAllocated
 Gibt die Speicheranzahl zurück, die im Computer-RAM verteilt sind

PDWORD pdw_ComputerBufferSizeArray
 Gibt die Größe der Speicher zurück, die Computer-RAM verteilt sind (in Byte).

Aufgabe:

Gibt die Speicherinformation der Karte und des vom Computer für diese Karte zugewiesenen Speichers zurück.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
DWORD dw_OnBoardRAMSize;
DWORD pdw_ComputerBufferSize[100];
BYTE    b_NbrOfComputerBuffer;
INT      i_ReturnValue;

```

```

i_ReturnValue = i_PCI3600_GetBoardInformation (dw_BoardHandle,
                                                &dw_OnBoardRAMSize,
                                                &b_NbrOfComputerBuffer,
                                                pdw_ComputerBufferSize);

```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch

7) i_PCI3600_CloseBoard 0**Syntax:**

`_INT_ i_PCI3600_CloseBoard(DWORD dw_BoardHandle)`

Parameter:**- Eingabe:**

`DWORD dw_BoardHandle` Handle der Karte

- Ausgabe:

Es erfolgt keine Ausgabe

Aufgabe:

Gibt den Handle der Karte frei. Verspermt den Zugang zur Karte.

Funktionsaufruf:

`DWORD dw_BoardHandle;`

`INT        i_ReturnValue;`

`i_ReturnValue = i_PCI3600_CloseBoard (dw_BoardHandle);`

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Die Funktion muss auf der Anwendungsebene aufgerufen werden.

8) i_PCI3600_SetBoardIntRoutine ()**Syntax:**

```

_INT_ i_PCI3600_SetBoardIntRoutine
    (BYTE b_BoardHandle,
     BYTE b_UserCallingMode,
     DWORD dw_UserSharedMemorySize,
     VOID ** ppv_UserSharedMemoryAppLevel,
     VOID ** ppv_UserSharedMemoryKernelLevel,
     VOID (WINAPI *v_FunctionName)
      (DWORD_ dw_BoardHandle,
       BYTE_ b_UserCallingMode,
       VOID * pv_UserSharedMemory,
       DWORD_ dw_InterruptSource,
       DWORD_ dw_InterruptMask,
       DWORD_ dw_NbrOfComputerBuffer,
       PDWORD_
pdw_ComputerBufferAddressKernelLevel,
       PDWORD_
pdw_ComputerBufferAddressApplicLevel))

```

Parameter:**- Eingabe:**

DWORD	dw_BoardHandle	Handle der Karte
BYTE	b_UserCallingMode	PCI3600_SYNCHRONOUS_MODE: Die Benutzer-Routine wird direct von der Interruptroutine des Treibers aufgerufen PCI3600_ASYNCHRONOUS_MODE: Die Benutzer-Routine wird vom Interrupt-Thread des Treibers aufgerufen.
DWORD	dw_UserSharedMemorySize	Legt die Größe (in Bytes) des Benutzer-Gemeinschaftsspeichers fest. Wird nur verwendet, wenn Sie Folgendes gewählt haben: PCI3600_SYNCHRONOUS_MODE
VOID	v_FunctionName	Name der Benutzer-Interruptroutine.

- Ausgabe

VOID **	ppv_UserSharedMemoryAppLevel	Adresse des Benutzer-Gemeinschaftsspeichers für die Anwendungsebene (Ring 3). Wird nur verwendet, wenn Sie Folgendes gewählt haben: PCI3600_SYNCHRONOUS_MODE
VOID **	ppv_UserSharedMemoryKernelLevel	Adresse des Benutzer-Gemeinschaftsspeichers für die Kernel-Ebene (Ring 0). Wird nur

verwendet, wenn Sie Folgendes gewählt haben:

PCI3600_SYNCHRONOUS_MODE

Aufgabe:

Falls Sie Visual Basic 5.0/6.0 verwenden:

- ist nur der asynchrone Mode ist verfügbar.

i

WICHTIG!

Windows 32-Bit Information

Für Windows NT/2000/XP und Windows 98 sind 4 Ringe (Ring 0 bis Ring 3) verfügbar.

- Die Anwendung des Benutzers arbeitet in Ring 3. Dieser Ring ermöglicht keinen Hardware-Zugriff.
- VXD- und SYS-Treiber arbeiten in Ring 0 und ermöglichen Zugriff auf Hardware.
- Ring 0 hat keinen direkten Zugriff zu der globalen Variablen von Ring 3. Er muss als Gemeinschaftsspeicher verwendet werden.
- Ring 0 und Ring 3 haben einen Zeiger, der auf diesen Gemeinschaftsspeicher zeigt. Die beiden Zeiger sind nicht unter derselben Adresse konfiguriert. Die Funktion muss für jede **APCI-3600**, für die ein Interrupt aktiviert werden soll, aufgerufen werden. Es wird eine Benutzer-Interrupt-Funktion auf allen Karten, auf denen ein Interrupt ausgelöst werden soll, installiert.

Erster Aufruf (erste Karte):

- die Benutzer-Interruptroutine wird installiert
- die Interrupts werden aktiviert
- der Benutzer-Gemeinschaftsspeicher wurde zugeordnet, wenn Folgendes ausgewählt wurde: PCI3600_SYNCHRONOUS_MODE

Falls Sie mehrere **APCI-3600** betreiben, die auf Interrupts reagieren sollen, rufen Sie die Funktion immer auf, wenn Sie die **APCI-3600** benutzen.

Die Variable *v_FunctionName* ist nur beim **ersten Aufruf von Bedeutung**.

Vom zweiten Funktionsaufruf an (nächste Karte) sind die Interrupts aktiviert

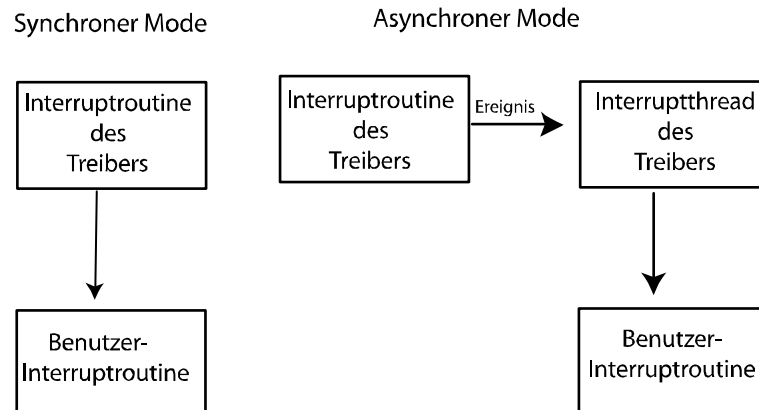
Interrupt

Die Benutzer-Interruptroutine wird vom System aufgerufen, wenn ein Interrupt erzeugt worden ist. Fall mehrere Karten betrieben werden und wenn diese auf Interrupts reagieren sollen, gibt die Variable *dw_BoardHandle* die Identifikationsnummer (Handle) der Karte zurück, die den Interrupt erzeugt hat.

Die Benutzer-Interruptroutine kann aufgerufen werden:

- d) direkt vom Treiber der Interruptroutine (synchroner Mode). Der Code der Benutzer-Interruptroutine arbeitet direct in Ring 0.
- e) vom Interrupt-Thread des Treibers (asynchroner Mode). Ein Ereignis wird ausgelöst and der Interrupt-Thread ruft die Benutzer-Interruptroutine auf. Der Code der Benutzer-Interruptroutine arbeitet in Ring 3.

Der Interrupt-Thread des Treibers hat im System höchste Priorität (31)

Abb. 9-1: Synchroner und asynchroner Mode**Tabelle 9-3: Synchroner Mode**

VORTEILE	Der Code der Benutzer-Interruptroutine wird direkt vom Treiber der Interruptroutine (Ring 0) aufgerufen. Die Zeit zwischen dem Interrupt und der Benutzer-Interruptroutine ist verkürzt.
	Der Benutzer kann die Benutzer-Interruptroutine nicht debuggen.
	Die Benutzer-Routine kann keine Windows API-Funktionen aufrufen.
GRENZEN	Die Benutzeroutine kann keine Funktionen aufrufen, die Zugriff auf globale Variablen geben. Der Benutzer kann weiterhin einen Gemeinschaftsspeicher verwenden.
	Dieser Mode ist nicht für Visual Basic verfügbar

Tabelle 9-4: Asynchroner Mode

VORTEILE	Der Benutzer kann die Benutzer-Interruptroutine debuggen, vorausgesetzt, dass er nicht in Visual Basic 5 programmiert hat
	Die Benutzer-Routine kann Windows API-Funktionen aufrufen.
	Die Benutzer-Routine kann Funktionen aufrufen, die Zugriff auf globale Variablen geben.
	Die Benutzer-Routine kann alle APCI-3600 Treiberfunktionen mit der folgenden Endung aufrufen: "i_PCI3600_XXXX"
GRENZEN	Der Code der Benutzer-Interruptroutine wird von der Interruptthreadroutine des Treibers aufgerufen (Ring 3). Die Zeit zwischen dem Interrupt und der Benutzer-Interruptroutine erhöht sich.

Gemeinschaftsspeicher

Wenn Sie den `PCI3600_SYNCHRONOUS_MODE` gewählt haben, haben Sie keinen Zugriff auf die globalen Variablen. Aber Sie haben die Möglichkeit, einen Gemeinschaftsspeicher (`ppv_UserSharedMemory`) einzurichten. Der Benutzer-Gemeinschaftsspeicher kann alle vordefinierten Compiler-Typen oder Benutzer-Definietypen enthalten.

Die Variable `dw_UserSharedMemorySize` gibt die Größe in Bytes des ausgewählten Benutzertyps an. Ein Zeiger der Variablen `ppv_UserSharedMemory` wird der Benutzer-Interruptroutine mit der Variablen `pv_UserSharedMemory` zugeteilt. Dies ist bei Visual Basic nicht möglich.

Die Benutzer-Interruptroutine muss folgende Syntax aufweisen:

```

VOID  v_FunctionName  (DWORD    dw_BoardHandle,
                       BYTE      b_UserCallingMode,
                       VOID *    pv_UserSharedMemory,
                       DWORD      dw_InterruptSource,
                       DWORD      dw_InterruptMask,
                       DWORD      dw_NbrOfComputerBuffer,
                       PDWORD
                       pdw_ComputerRAMAddressKernelLevel,
                       PDWORD
                       pdw_ComputerRAMAddressApplicLeve)

```

<code>v_FunctionName</code>	Name der Benutzer-Interruptroutine
<code>dw_BoardHandle</code>	Handle der APCI-3600 , die den Interrupt generiert hat.
<code>b_UserCallingMode</code>	<code>PCI3600_SYNCHRONOUS_MODE</code> : Die Benutzer-Routine wird direkt von der Interrupt-Routine des Treibers aufgerufen

	PCI3600_ASYNCHRONOUS_MODE:
	Die Benutzer-Routine wird vom Interrupt-Thread des Treibers aufgerufen.
<i>pv_UserSharedMemory</i>	Zeiger des Benutzer-Gemeinschaftsspeichers
<i>dw_InterruptSource</i>	Quelle des Interrupts
<i>dw_InterruptMask</i>	Maske der Events, die den Interrupt erzeugt haben.
<i>dw_NbrOfComputerBuffer</i>	Anzahl der PC-Speicher, welche die Werte enthalten.
<i>pdw_ComputerRAMAddressKernelLevel</i>	Array der Adresse des Speichers, der die Werte für die Kernel-Ebene enthält (Ring 0)
<i>pdw_ComputerRAMAddressAppLevel</i>	Array der Adresse des Speichers, der die Werte für die Anwendungsebene enthält (Ring 3)

Tabelle 9-5: dw - Interruptquelle

Wert	Bedeutung
0000 0000 0000 0000 0000 0000 0000 0001	Analoger Ausgang 0
0000 0000 0000 0000 0000 0000 0000 0010	Analoger Ausgang 1
0000 0000 0000 0000 0000 0000 0000 0100	Analoges Eingangsmodul 0
0000 0000 0000 0000 0000 0000 0000 1000	Analoges Eingangsmodul 1
0000 0000 0000 0000 0000 0000 0001 0000	Analoges Eingangsmodul 2
0000 0000 0000 0000 0000 0000 0010 0000	Analoges Eingangsmodul 3
0000 0000 0000 0000 0000 0000 0100 0000	Chronometer 0
0000 0000 0000 0000 0000 0000 1000 0000	Chronometer 1
0000 0000 0000 0000 0000 0001 0000 0000	Chronometer 2
0000 0000 0000 0000 0000 0010 0000 0000	Chronometer 3

Tabelle 9-6: dw - Interruptmaske

Funktion	Wert	Bedeutung
Analoger Ausgang	0000 0000 0000 0000 0000 0000 0000 0001	Vergleicht, ob ein Interrupt ausgelöst wurde und ein Transfer erfolgt ist.
	0000 0000 0000 0000 0000 0000 0000 0010	Interner FIFO-Fehler-Interrupt
	0000 0000 0000 0000 0000 0000 0000 0100	SDRAM Underflow-Interrupt
	0000 0000 0000 0000 0000 0000 0000 1000	PC-Speicher Underflow-Interrupt
Analoges Eingangsmodul / Chronometer	0000 0000 0000 0000 0000 0000 0000 0001	DMA-Interrupt
	0000 0000 0000 0000 0000 0000 0000 0010	Interner FIFO-Fehler-Interrupt
	0000 0000 0000 0000 0000 0000 0000 0100	SDRAM Overflow-Interrupt
	0000 0000 0000 0000 0000 0000 0000 1000	PC-Speicher Overflow-Interrupt
	0000 0000 0000 0000 0000 0000 0001 0000	Auto Stop-Interrupt

Tabelle 9-7: Speicher

Der in Parametern angegebene Speicher in der Interrupt-Routine hat das folgende Format:

Address	Write/Read																																							
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0								
Address + 0	Number of values																												Index		Source									
Address + 4	Value 0																																							
Address + 8	Value 1																																							
Address + 12	...																																							

“Source“: Quelle des Speichers

0 : ADC

1 : Chronometer

„Index“: Index der Quelle

0 : ADC/Chronometer 0

1 : ADC/Chronometer 1

...

„Number of values“: Anzahl der Werte, die in den Speicher transferiert wurden.

```

BYTE      b_UserCallingMode,
DWORD     dw_UserSharedMemorySize,
VOID **   ppv_UserSharedMemoryApp,
VOID **   ppv_UserSharedMemoryKernel,
VOID      v_FunctionName(DWORD     dw_BoardHandle,
                             BYTE     b_UserCallingMode,
                             VOID *    pv_UserSharedMemory,
                             DWORD     dw_InterruptMask,
                             DWORD     dw_BufferHandle,
                             DWORD     dw_NbrOfComputerBuffer,
                             PDWORD    pdw_ComputerRAMAddressKernelLevel,
                             PDWORD    pdw_ComputerRAMAddressApplicLevel)

```

Funktionsaufruf:

```

typedef struct
{
    .
    .
    .
}str_UserStruct;

str_UserStruct * ps_UserSharedMemory;

void    v_FunctionName    (unsigned long  dw_BoardHandle,
                          unsigned char  b_UserCallingMode,
                          void *        pv_UserSharedMemory,
                          unsigned long  dw_InterruptSource,
                          unsigned long  dw_InterruptMask,
                          unsigned long  dw_NbrOfComputerBuffer,
                          unsigned long
                          *pdw_ComputerRAMAddressKernelLevel,
                          unsigned long
                          *pdw_ComputerRAMAddressApplicLevel)
    {
        str_UserStruct * ps_InterruptSharedMemory;

        ps_InterruptSharedMemory = (str_UserStruct *)
pv_UserSharedMemory;
        .
        .
    }

DWORD dw_BoardHandle;
INT    i_ReturnValue;

i_ReturnValue = i_PCI3600_SetBoardIntRoutine (dw_BoardHandle,

PCI3600_SYNCHRONOUS_MODE,

                                sizeof (str_UserStruct),
                                (void **))

&ps_UserSharedMemoryApp,

                                (void **)

&ps_UserSharedMemoryKernel,

                                v_FunctionName);

```

Return-Wert:

- 0: Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2: Diese Funktion kann nicht in der Kernel-Ebene aufgerufen werden.
- 3 : Falscher Aufrufname
- 4 : Kein freier Speicherplatz für den globalen Speicher des Benutzers
- 5: Die Interrupt-Verwaltung wird von diesem Compiler nicht unterstützt

- 6: Die Wahl des Aufruf-Modes der Benutzer-Interruptroutine war falsch
- 7: Interrupt is schon installiert
- 8: Interrupt hat keine Eigenschaft
- 9: ACPI aktiv, aber Windows NT 4 wird verwendet.
- 10: PNP OS aktiv, aber Windows NT 4 wird verwendet
- 11: Fehler beim Erzeugen des Benutzer-Gemeinschaftsspeichers.
- 12: Bereitet die Benutzerfunktion auf den Kernelausführungsfehler vor.
- 13: Fehler beim Erzeugen eines API-Interruptereignisses
- 14: Fehler beim Erzeugen einer API Interruptthreadfunktion
- 15: Fehler beim Erzeugen eines Rückrufs der API-Funktion
- 16: Fehler beim Installieren einer API Interruptfunktion
- 17: Fehler beim Erzeugen eines Interruptparameters im Gemeinschaftsspeicher
- 18: Fehler beim Vorbereiten eines wichtigen Abschnitts.

9) i_PCI3600_ResetBoardIntRoutine ()**Syntax:**

INT i_PCI3600_ResetBoardIntRoutine(DWORD dw_BoardHandle)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle oder Karte

Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Stoppt die Interruptverwaltung der Karte **APCI-3600**.

Deinstalliert die Benutzer-Interruptroutine, wenn die Interruptverwaltung aller **APCI-3600** – Karten gestoppt wurde.

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

i_ReturnValue = i_PCI3600_ResetBoardIntRoutine (dw_BoardHandle);

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Diese Funktion kann nicht in der Kernel-Ebene aufgerufen werden

-3: Die Interruptroutine ist nicht installiert

10) i_PCI3600_Write32BitPortValue()**Syntax:**

```
_INT_ i_PCI3600_Write32BitPortValue(DWORD dw_BoardHandle,  
                                     BYTE   b_AddressSelection,  
                                     DWORD dw_Offset,  
                                     DWORD dw_Value)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_AddressSelection	Auswahl der Adresse der Karte mit Zugriff.
DWORD dw_Offset	Offset, bei dem der Wert geschrieben werden muss.
DWORD dw_Value	Zu schreibender Wert.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Führt einen 32-Bit Ausgangszugriff (E/A oder Speicher) durch beim Offset *dw_Offset* von der Adresse *b_AddressSelection*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT     i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_Write32BitPortValue (dw_BoardHandle,  
                                                0,  
                                                0,  
                                                0xFFF0000FUL);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2 : Falsche Wahl der Adresse
-3: Ausgewählter Offset ist nicht verfügbar

11) i_PCI3600_Read32BitPortValue()**Syntax:**

```
_INT_ i_PCI3600_Read32BitPortValue(DWORD dw_BoardHandle,  
                                     BYTE   b_AddressSelection,  
                                     DWORD dw_Offset,  
                                     PDWORD pdw_Value)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_AddressSelection	Auswahl der Adresse der Karte mit Zugriff
DWORD dw_Offset	Offset, bei dem der Wert gelesen werden muss.

- Ausgabe

PDWORD pdw_Value	Wert lesen.
------------------	-------------

Aufgabe:

Führt einen 32-Bit-Zugriff (E/A oder Speicher) auf der Karte beim Offset *dw_Offset* von der Adresse *b_AddressSelection* durch.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT     i_ReturnValue;  
DWORD dw_Value;
```

```
i_ReturnValue = i_PCI3600_Read32BitPortValue (dw_BoardHandle,  
                                              0,  
                                              0,  
                                              &dw_Value);
```

Return-Wert:

0 : Kein Fehler
-1-: Der Handle-Parameter der Karte ist falsch
-2 : Falsche Wahl der Adresse
-3 : Ausgewählter Offset ist nicht verfügbar

9.1.2 Analoge Eingänge

12) i_PCI3600_InitAnalogInputModule()

Syntax:

```

_INT_ i_PCI3600_InitAnalogInputModule(DWORD dw_BoardHandle,
                                         BYTE      b_Module,
                                         PBYTE     pb_SingleDiffMode,
                                         PBYTE     pb_Coupling,
                                         PBYTE     pb_EnableCurrentSource,
                                         PBYTE     pb_GainSelection,
                                         BYTE      b_SamplingClockSelection,
                                         BYTE      b_SpeedMode)

```

Parameter:

-Eingabe:

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangsmoduls (0 bis 3)
PBYTE pb_SingleDiffMode	Single-Ended / Differentieller Mode eines jeden Kanals pb_SingleDiffMode[0] : Linker Kanal pb_SingleDiffMode[1] : Rechter Kanal 0 : Single-Ended 1 : Differentiell
PBYTE pb_Coupling	Coupling-Mode eines jeden Kanals pb_Coupling [0] : Linker Kanal pb_Coupling [1] : Rechter Kanal 0 : DC 1 : AC
PBYTE pb_EnableCurrentSource	Aktiviere/deaktiviere die aktuelle Quelle eines jeden Kanals. pb_EnableCurrentSource[0] : Linker Kanal pb_EnableCurrentSource[1] : Rechter Kanal 0 : Deaktiviere 1 : Aktiviere
PBYTE pb_GainSelection	Auswahl des Gains pb_GainSelection[0] : Linker Kanal pb_GainSelection[1] : Rechter Kanal 0 : Gain 1 1 : Gain 10
BYTE b_SamplingClockSelection	Auswahl des Sampling-Clocks für das analoge Eingangsmodul 0 : Sampling Clock 1 1 : Sampling Clock 2
BYTE b_SpeedMode	ADC Auswahl des Speed Modes 00 : Single speed mode (2KHz – 50KHz) 01 : Double speed mode (50KHz – 100KHz)

10 : Quadruple speed mode (100KHz – 200KHz)

Tabelle 9-8: ADC-Clocks

			Single Speed	Double Speed	Quad Speed
			00	01	10
Master Clock (MHz)	Divisor-Faktor	Clock-Generator 1/2	Abtast-frequenz (Hz)	Abtast-frequenz (Hz)	Abtast-frequenz (Hz)
102,4	4	25600000	50000	100000	200000
	5	20480000	40000	80000	160000
	6	17066667	33333	66667	133333
	8	12800000	25000	50000	100000
	10	10240000	20000		
	12	8533333	16667		
	16	6400000	12500		
	20	5120000	10000		
	25	4096000	8000		
	40	2560000	5000		
	50	2048000	4000		
	60	1706667	3333		
	80	1280000	2500		
	100	1024000	2000		

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert das analoge Eingangsmodul *b_Module*.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT     i_ReturnValue;
BYTE  pb_SingleDiffMode[2];
BYTE  pb_Coupling[2];
BYTE  pb_EnableCurrentSource[2];
BYTE  pb_EnableCurrentSource[2];
BYTE  pb_GainSelection[2];

```

```

i_ReturnValue = i_PCI3600_InitAnalogInputModule(dw_BoardHandle,
0,

```

```
pb_SingleDiffMode,  
pb_Coupling,  
pb_EnableCurrentSource,  
pb_GainSelection,  
0,
```

```
PCI3600_SAMPLING_CLOCK_1,
```

```
PCI3600_SINGLE_SPEED_MODE);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2 : Falsche Modulnummer

-3 : Falsche Wahl des Single/Diff. Modes

-4 : Falsche Auswahl des Couplings

-5 : Falsche Wahl des Flags der aktuellen Quelle

-6 : Falsche Wahl des Gains

-7 : Falsche Wahl des Clocks

-8 : Falscher Speed Mode-Parameter

-9: Die Karte ist nicht initialisiert

-10: Speed Mode ist nicht verfügbar

-11: Die aktuelle Quelle ist nur auf den ersten beiden Modulen verfügbar

-12: Wenn die aktuelle Quelle verwendet wird, muss der Kanal Single-Ended und AC konfiguriert sein

-13: Das Modul wurde schon von einem anderen Prozess initialisiert.

-14: Initialisierungsfehler

14) i_PCI3600_GetAnalogInputModuleCalibrationStatus()**Syntax:**

```
_INT_ i_PCI3600_GetAnalogInputModuleCalibrationStatus (DWORD  
dw_BoardHandle,  
  
b_Module,  
  
pb_Status)
```

BYTE
PBYTE

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangmoduls (0 bis 3)

- Ausgabe:

PBYTE pb_Status	Status der Kalibrierung
	1: Kalibrierung läuft
	0: Ende der Kalibrierung: Das Modul ist bereit

Aufgabe:

Status der Kalibrierung des analogen Eingangmoduls *b_Module*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;  
BYTE b_Status;  
  
i_ReturnValue =  
i_PCI3600_GetAnalogInputModuleCalibrationStatus(dw_BoardHandle,  
0,  
  
&b_Status);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Modulnummer

15) i_PCI3600_StartAnalogInputModuleSingleAcquisition()**Syntax:**

`_INT_ i_PCI3600_StartAnalogInputModuleSingleAcquisition(DWORD
dw_BoardHandle,

BYTE
b_Module)`

Parameter:**- Eingabe:**

DWORD	dw_BoardHandle	Handle der Karte
BYTE	b_Module	Index des analogen Eingangmoduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Start einer Einzelerfassung auf dem analogen Eingangsmodul *b_Module* .

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue =
i_PCI3600_StartAnalogInputModuleSingleAcquisition(dw_BoardHandle,
0);

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2 : Falsche Modulnummer

-3: Das Modul ist nicht initialisiert

-4: Das Modul ist für den zyklischen Erfassungsmodus initialisiert

-5: Eine Konvertierung wurde schon gestartet

-6: Die Kalibrierung läuft auf dem Modul

16) i_PCI3600_GetAnalogInputModuleSingleAcquisitionStatus()**Syntax:**

```
_INT_ i_PCI3600_GetAnalogInputModuleSingleAcquisitionStatus  
      (DWORD dw_BoardHandle,  
       BYTE   b_Module,  
       PBYTE  pb_Status)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle of the board
BYTE b_Module	Index of the analog input module (0 to 3)

- Ausgabe

PBYTE pb_Status	Status of the single acquisition (equal to the status of the FIFO)
	0: Value not available
	1: Value available

Aufgabe:

Gibt den Status der Einzelerfassung zurück

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT    i_ReturnValue;  
BYTE   b_Status;
```

```
i_ReturnValue = i_PCI3600_GetAnalogInputModuleSingleAcquisitionStatus  
                (dw_BoardHandle,  
                 0,  
                 &b_Status);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Modulnummer
-3: Das Modul ist nicht initialisiert
-4: Die Kalibrierung läuft auf dem Modul

17) i_PCI3600_ReadAnalogInputSingleAcquisitionValues()**Syntax:**

```

_INT_ i_PCI3600_ReadAnalogInputSingleAcquisitionValues
        (DWORD dw_BoardHandle,
         BYTE b_Module,
         PDWORD pdw_Values)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangmoduls (0 bis 3)

- Ausgabe

PDWORD pdw_Values	Values of the single acquisition
	pdw_Values[0] : Value of the left channel
	pdw_Values[1] : Value of the right channel

Aufgabe:

Liest den Wert des rechten/linken Kanals.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT i_ReturnValue;
DWORD pdw_Values[2];

```

```

i_ReturnValue = i_PCI3600_ReadAnalogInputSingleAcquisitionValues
(dw_BoardHandle,

```

```

0,

```

```

pdw_Values);

```

Return-Wert:

- 0 : Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch.
- 2: Falsche Modulnummer
- 3: Das Modul ist nicht initialisiert
- 4: Die Werte sind nicht verfügbar
- 5: Die Kalibrierung läuft auf dem Modul

18) i_PCI3600_GetAnalogInputModuleValues()**Syntax:**

```
_INT_ i_PCI3600_GetAnalogInputModuleValues(DWORD dw_BoardHandle,  
                                             BYTE      b_Module,  
                                             PDWORD pdw_Values)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Inputmoduls (0 bis 3)

- Ausgabe

PDWORD pdw_Values	Werte der Erfassung
pdw_Values[0]	: Wert des linken Kanals
pdw_Values[1]	: Wert des rechten Kanals

Aufgabe:

Startet und liest den Wert des rechten/linken Kanals des analogen Eingangsmoduls *b_Module*.

Die Funktion wartet darauf, dass die Werte zum Lesen bereitstehen.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;  
DWORD pdw_Values[2];
```

```
i_ReturnValue = i_PCI3600_GetAnalogInputModuleValues(dw_BoardHandle,  
                                                       0,  
                                                       pdw_Values);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Modulnummer

-3: Das Modul ist nicht gestartet-

-4: Fehler beim Starten des analogen Eingangsmoduls

-5: Zeitfunktionsfehler

-6: Fehler beim Lesen des Status des analogen Eingangsmoduls

-7: Fehler beim Lesen der Werte des analogen Eingangsmoduls

-8: Time out tritt ein

19) i_PCI3600_InitAnalogInputModuleCyclicAcquisition()

Syntax:

```
_INT_ i_PCI3600_InitAnalogInputModuleCyclicAcquisition
        (DWORD dw_BoardHandle,
         BYTE   b_Module,
         BYTE   b_Mode,
         DWORD dw_OnBoardBufferSize,
         DWORD dw_CompareValue)
```

Parameter:

- Eingabe:

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangsmoduls (0 bis 3)
BYTE b_Mode	Definiert den verwendeten Mode
	0: AUTO STOP-Mode
	In diesem Mode ist der Speicher auf der Karte mit Werten gefüllt.
	Wenn der On board-Speicher noch voll ist, wird die Erfassung gestoppt und ein Interrupt wird ausgelöst.
	1: RING Buffer Mode
	In diesem Mode ist der On Board-Speicher immer mit Werten gefüllt.
	In beiden Modes: Immer, wenn der Wert im Speicher erreicht ist dw_CompareValue, wird ein Interrupt ausgelöst, damit die Software den Wert vom Speicher lesen kann
DWORD dw_OnBoardBufferSize	Größe (in nbr of bytes : $1024 * 2$ dw_OnBoardBufferSize) des Speichers auf der Karte (im on-board RAM), das für das analoge Eingangsmodul verwendet wird.
DWORD dw_CompareValue	Definiert die Anzahl der Erfassungen, die im Speicher auf der Karte erfolgen soll, um einen Interrupt zu erzeugen zum Start eines DMA-Transfers.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert die zyklische Erfassung des analogen Eingangsmoduls *b_Module*.

Tabelle 9-9: RAM-Speichergröße auf der Karte

dw_OnBoardBufferSize	Nbr of values
0	256
1	512
2	1024
3	2048
..	...

Funktionsaufruf:

DWORD dw_BoardHandle;
 INT i_ReturnValue;

```
i_ReturnValue = i_PCI3600_InitAnalogInputModuleCyclicAcquisition
(dw_BoardHandle,
0,
PCI3600_AUTO_STOP_MODE,
0,
10);
```

Return-Wert:

- 0: Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2: Falsche Modulnummer
- 3: Falscher Parameter-Mode
- 4: Das Modul ist nicht initialisiert
- 5: Es ist kein PC-Speicher verfügbar.
- 6: Der Vergleichwert kann Folgendes nicht sein:
 - größer als die Speichergröße auf der Karte im Auto Stop-Mode
 - größer als oder gleich wie die Speichergröße auf der Karte im Ring Buffer Mode
 - Null
- 7: Die zyklische Erfassung wurde schon von einem anderen Prozess initialisiert.
- 8: Die zyklische Erfassung läuft.
- 9: Nicht genügend Platz auf dem RAM auf der Karte
- 10: Die Interruptroutine ist nicht installiert
- 11: Primary Init Error
- 12: Die Kalibrierung läuft auf dem Modul

20) i_PCI3600_StartAnalogInputModuleCyclicAcquisition()**Syntax:**

```
_INT_ i_PCI3600_StartAnalogInputModuleCyclicAcquisition(DWORD  
dw_BoardHandle,  
  
b_Module)
```

BYTE

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingabemoduls (0 bis 3).

- Ausgabe

Keine Ausgabe

Aufgabe:Startet die analoge zyklische Erfassung für das Modul *b_Module*.**Funktionsaufruf:**

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_StartAnalogInputModuleCyclicAcquisition  
(dw_BoardHandle,  
  
0);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Modulnummer

-3: Die zyklische Erfassung ist nicht initialisiert oder nicht von diesem Prozess initialisiert

21) i_PCI3600_GetAnalogInputModuleCyclicAcquisitionStatus()**Syntax:**

```
_INT_ i_PCI3600_GetAnalogInputModuleCyclicAcquisitionStatus
      (DWORD dw_BoardHandle,
       BYTE b_Module,
       PBYTE pb_Status,
       PDWORD pdw_NbrOfValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangsmoduls des analogen Eingangsmoduls (0 bis 3)

- Ausgabe

PBYTE pb_Status	Status der zyklischen Erfassung 0: nicht gestartet 1: gestartet
PDWORD pdw_NbrOfValue	Anzahl der Werte im Speicher des RAM auf der Karte für das Modul

Aufgabe:

Gibt den Status und die Anzahl der Werte im RAM der Karte für das Modul *b_Module* zurück.

Funktionsaufruf:

```
DWORD dw_BoardHandle;
INT i_ReturnValue;
BYTE b_Status;
DWORD dw_NbrOfValue;
```

```
i_ReturnValue = i_PCI3600_GetAnalogInputModuleCyclicAcquisitionStatus
(dw_BoardHandle,
                                0,
    &b_Status,
    &dw_NbrOfValue);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Modulnummer
-3: Das Modul ist nicht initialisiert

22) i_PCI3600_IncrementAnalogInputCyclicAcquisitionBufferReadPointer()**Syntax:**

```
_INT_ i_PCI3600_IncrementAnalogInputCyclicAcquisitionBufferReadPointer  
      (DWORD dw_BoardHandle,  
       DWORD dw_NbrOfValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
DWORD dw_NbrOfValue	Anzahl der Werte

Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Ermöglicht die Inkrementierung des Lesezeigers des Ring Buffers, der für den DMA-Transfer benötigt wird.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT      i_ReturnValue;  
DWORD dw_BufferHandle;
```

```
i_ReturnValue =  
i_PCI3600_IncrementAnalogInputCyclicAcquisitionBufferReadPointer  
  
(dw_BoardHandle,  
  
  
1000);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch

23) i_PCI3600_StopAnalogInputModuleCyclicAcquisition()**Syntax:**

`_INT_ i_PCI3600_StopAnalogInputModuleCyclicAcquisition(DWORD
dw_BoardHandle,

BYTE
b_Module)`

Parameter:**-Eingabe:**

DWORD dw_BoardHandle	Handle of the board
BYTE b_Module	Index of the analog input module (0 to 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Stoppt die zyklische analoge Eingangserfassung für das Modul *b_Module*.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

`i_ReturnValue = i_PCI3600_StopAnalogInputModuleCyclicAcquisition
(dw_BoardHandle,

0);`

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Modulnummer

-3: Die zyklische Erfassung wurde nicht gestartet oder wurde von einem anderen
Prozess gestartet.

24) i_PCI3600_ReleaseAnalogInputModuleCyclicAcquisition()**Syntax:**

```
_INT_ i_PCI3600_ReleaseAnalogInputModuleCyclicAcquisition  
                                     (DWORD dw_BoardHandle,  
                                     BYTE      b_Module)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangmoduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt die zyklische analoge Eingangserfassung für das Modul *b_Module* frei.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT     i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_ReleaseAnalogInputModuleCyclicAcquisition  
                                     (dw_BoardHandle,  
                                     0);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Modulnummer
-3: Das Modul ist nicht initialisiert

25) i_PCI3600_AnalogInput_EnableDisableDigitalInputExternalTrigger()**Syntax:**

```

_INT_ i_PCI3600_AnalogInput_EnableDisableDigitalInputExternalTrigger
        (DWORD dw_BoardHandle,
         BYTE    b_Module,
         BYTE    b_ExternTriggerFlag)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangsmoduls (0 bis 3)
BYTE b_ExternTriggerFlag	Aktiviert/deaktiviert das Flag für den externen Trigger der digitalen Eingänge 0 : Deaktivieren 1 : Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den externen Trigger der digitalen Eingänge (digitaler Eingang 0) für das analoge Eingangsmodul *b_Module*.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT    i_ReturnValue;

```

```

i_ReturnValue =
i_PCI3600_AnalogInput_EnableDisableDigitalInputExternalTrigger
        (dw_BoardHandle,
         0,
         PCI3600_ENABLE);

```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Modulnummer
-3: Flag des externen Triggers ist falsch
-4: Das Modul ist nicht initialisiert
-5: Der digitale externe Trigger ist nicht initialisiert

9.1.3 Analoge Ausgänge

26) i_PCI3600_InitAnalogOutputChannel()

Syntax:

```

_INT_ i_PCI3600_InitAnalogOutputChannel(DWORD dw_BoardHandle,
                                         BYTE    b_Channel,
                                         BYTE
b_SamplingClockSelection,
                                         BYTE    b_SpeedMode)

```

Parameter:

- Eingabe:

DWORD dw_BoardHandle Handle der Karte
 BYTE b_Channel Index des analogen Eingangskanals (0 bis 1)
 BYTE b_SamplingClockSelection
 Clock-Auswahl für den analogen
 Ausgangskanal
 0 : Sampling-Clock 1
 1 : Sampling Clock 2
 BYTE b_SpeedMode DAC Auswahl des Speed Modes
 00 : Single Speed Mode (2kHz – 50kHz)
 01 : Double Speed Mode (50kHz – 100kHz)
 10 : Quadruple Speed Mode (100kHz –
 200kHz)

Tabelle 9-10: DAC Sampling-Clock

DAC Sampling Clock					
Master Clock (MHz)	Divisor	Clock Gen ½ (Hz)	Single Speed	Double Speed	Quad Speed
			00	01	10
			Abtastfrequenz (Hz)	Abtastfrequenz (Hz)	Abtastfrequenz (Hz)
102,4	4	25600000	50000	100000	200000
	5	20480000	40000	80000	160000
	6	17066667	33333	66667	133333
	8	12800000	25000	50000	100000
	10	10240000	20000		
	12	8533333	16667		
	16	6400000	12500		
	20	5120000	10000		
	25	4096000	8000		
	40	2560000	5000		
	50	2048000	4000		
	60	1706667	3333		
	80	1280000	2500		
	100	1024000	2000		

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert den analogen Ausgangskanal *b_Channel*.

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

```
i_ReturnValue = i_PCI3600_InitAnalogOutputChannel  
                (dw_BoardHandle,  
                 0,  
                 PCI3600_SAMPLING_CLOCK_1,  
                 PCI3600_SINGLE_SPEED_MODE);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Falsche Auswahl des Clocks

-4: Falscher Speed Mode-Parameter

-5: Die Karte ist nicht initialisiert

-6: Speed Mode ist nicht verfügbar

-7: Der Kanal wurde schon von einem anderen Prozess initialisiert

-8: Initialisierungsfehler

27) i_PCI3600_ReleaseAnalogOutputChannel()**Syntax:**

`_INT_ i_PCI3600_ReleaseAnalogOutputChannel(DWORD dw_BoardHandle,
 BYTE b_Channel)`

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle of the board
BYTE b_Channel	Index of the analog output channel (0 to 1)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt den analogen Ausgangskanal *b_Channel* frei.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

`i_ReturnValue = i_PCI3600_ReleaseAnalogOutputChannel (dw_BoardHandle,
 0);`

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Kanal ist nicht initialisiert oder wird von einem anderen Prozess
initialisiert

28) i_PCI3600_GetAnalogOutputReadyBitStatus()**Syntax:**

```
_INT_ i_PCI3600_GetAnalogOutputReadyBitStatus(DWORD dw_BoardHandle,  
                                                BYTE    b_Channel,  
                                                PBYTE  
pb_ReadyBitStatus)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

PBYTE pb_ReadyBitStatus	Status des fertigen Bits für den analogen Ausgangskanals. 0: Kanal nicht bereit für den Empfang eines Wertes 1: Kanal bereit zum Empfang eines Wertes
-------------------------	---

Aufgabe:

Gibt den Status des fertigen Bits zurück, um zu erfahren, ob ein Wert auf den analogen Ausgangskanal *b_Channel* geschrieben werden kann.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT    i_ReturnValue;  
BYTE   b_ReadyBitStatus;
```

```
i_ReturnValue = i_PCI3600_GetAnalogOutputReadyBitStatus (dw_BoardHandle,  
                                                         0,
```

```
&b_ReadyBitStatus);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Kanalnummer
-3: Der Kanal list nicht initialisiert
-4: Der Kanal ist für den Signalgeneratormode initialisiert

29) i_PCI3600_WriteAnalogOutputValue()**Syntax:**

```
_INT_ i_PCI3600_WriteAnalogOutputValue(DWORD dw_BoardHandle,  
                                         BYTE    b_Channel,  
                                         WORD    w_Value)
```

Parameter:**- Eingabe**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Eingangskanals (0 bis 1)
WORD w_Value	Wert, der auf den analogen Ausgang zu schreiben ist (0 bis 65535)

- Ausgabe

Es erfolgt keine Ausgabe.

Aufgabe:

Schreibt den Wert *w_Value* auf den analogen Ausgang *b_Channel*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT    i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_WriteAnalogOutputValue (dw_BoardHandle,  
                                                    0,  
                                                    16383);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Kanal ist nicht initialisiert

-4: Diese Funktion kann nicht arbeiten, wenn der digitale externe Trigger aktiviert ist.

-5: Der Kanal ist für den Signalgeneratormode initialisiert

-6: Der Kanal ist nicht für den Empfang eines neuen Wertes bereit.

30) i_PCI3600_Set1AnalogOutputChannel()

Syntax:

INT i_PCI3600_Set1AnalogOutputChannel (DWORD dw_BoardHandle,
 BYTE b_Channel,
 WORD w_Value)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Eingangskanals (0 bis 1)
WORD w_Value	Wert, der auf den analogen Ausgang geschrieben werden soll (0 bis 65535)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt den analogen Ausgangskanal *b_Channel* auf den Wert *w_Value*. Das fertige Bit wird in der Funktion getestet.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_Set1AnalogOutputChannel (dw_BoardHandle,
 0,
 16383);

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Kanal ist nicht initialisiert

-4: Diese Funktion kann nicht arbeiten, wenn der digital externe Trigger aktiviert ist.

-5: Der Kanal ist für den Signalgeneratormode initialisiert

-6: Zeitfunktionsfehler

-7: Time out tritt ein

31) i_PCI3600_ReserveAnalogOutputSignalGeneratorComputerBuffer()**Syntax:**

```
_INT_ i_PCI3600_ReserveAnalogOutputSignalGeneratorComputerBuffer  
      (DWORD dw_BoardHandle,  
       BYTE   b_Channel,  
       DWORD  dw_BufferSize)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)
DWORD dw_BufferSize	Größe (in nbr der Erfassung) des Speichers, der für das analoge Ausgangsmodul verwendet wird.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Reserviert einen Teil des PC-Speichers, der von ADDIREG für den analogen Ausgang *b_Channel* zugeteilt wurde.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT     i_ReturnValue;
```

```
i_ReturnValue =  
i_PCI3600_ReserveAnalogOutputSignalGeneratorComputerBuffer  
  (dw_BoardHandle,  
   0,  
   1000);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Fehler Speichergröße

-4: Der Speicher ist schon von einem anderen Prozess reserviert

-5: Der PC DMA-Speicher wird verwendet

-6: Der Kanal ist im Signalgeneratormode initialisiert

32) i_PCI3600_FreeAnalogOutputSignalGeneratorComputerBuffer()**Syntax:**

```
_INT_ i_PCI3600_FreeAnalogOutputSignalGeneratorComputerBuffer  
                                     (DWORD  
dw_BoardHandle,  
                                     BYTE    b_Channel)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt den Teil des PC-Speichers, der von ADDIREG für den analogen Ausgang *b_Channel* zugeteilt wurde, frei.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT    i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_FreeAnalogOutputSignalGeneratorComputerBuffer  
                                     (dw_BoardHandle,  
                                     0);
```

Return -Wert:

- 0 : Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2: Falsche Kanalnummer
- 3: Der Speicher ist nicht belegt oder wird von einem anderen Prozess belegt
- 4: Der Kanal ist im Signalgeneratormode initialisiert

33) i_PCI3600_WriteAnalogOutputSignalGeneratorComputerBufferValues()**Syntax:**

```

_INT_ i_PCI3600_WriteAnalogOutputSignalGeneratorComputerBufferValues
    (DWORD
    dw_BoardHandle,
    BYTE    b_Channel,
    DWORD
    dw_NbrOfValue,
    PWORD   pw_ValueArray,
    PDWORD
    pdw_NbrOfWrittenValues,
    PDWORD
    pdw_NbrOfFreeValues)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)
DWORD dw_NbrOfValue	Anzahl der Werte, die in den Speicher geschrieben werden sollen
PWORD pw_ValueArray	Der Wert, der in den PC-RAM-Speicher geschrieben werden soll

- Ausgabe

PDWORD pdw_NbrOfWrittenValues	Anzahl der Werte, die in den PC-Speicher geschrieben wurden.
PDWORD pdw_NbrOfFreeValues	Anzahl der Werte, die noch in den PC-Speicher geschrieben werden können.

Aufgabe:

Schreibt die Werte des analogen Ausgangs *b_Channel* für diesen Kanal in den PC-RAM-Speicher.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT    i_ReturnValue;
WORD   pw_ValueArray[200];
DWORD  dw_NbrOfWrittenValues;
DWORD  dw_NbrOfFreeValues;

```

```

i_ReturnValue =
i_PCI3600_WriteAnalogOutputSignalGeneratorComputerBufferValues
    (dw_BoardHandle,
    0,
    100,
    pw_ValueArray,
    &dw_NbrOfWrittenValues,
    &dw_NbrOfFreeValues);

```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Speicher ist nicht belegt.

-4: Der Speicher ist voll

34) i_PCI3600_InitAnalogOutputSignalGenerator()**Syntax:**

```

_INT_ i_PCI3600_InitAnalogOutputSignalGenerator(DWORD
dw_BoardHandle,
                                                    BYTE    b_Channel,
                                                    BYTE    b_Mode,
                                                    DWORD
dw_OnBoardBufferSize,
                                                    DWORD
dw_CompareValue)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)
BYTE b_Mode	Mode des analogen Ausgangskanals
	0: Free Run Mode
	In diesem Mode wird der RAM-Speicher auf der Karte einmal gefüllt und dann wird dieser Wert immer dem analogen Ausgangskanal zugewiesen. Die reservierte PC-Speichergröße für diesen analogen Ausgang muss mit der Speichergröße auf der Karte übereinstimmen. In diesem Mode kann kein Underrun-Fehler eintreten.
	1: Ring Buffer Mode
	In diesem Mode wird der RAM-Speicher auf der Karte vor dem Start gefüllt und dann wird der Wert dem analogen Ausgangskanal gegeben. Wenn der Mindestwert erreicht ist, wird ein Interrupt ausgelöst und die Software lädt den Speicher mit neuen Werten des PC-Speichers (der Benutzer lädt mit folgender Funktion neue Werte in den PC-Speicher: i_PCI3600_WriteAnalogOutputSignalGeneratorComputerBufferValues(...)). Wenn der lokale DMA-Transfer zu schnell ist, tritt ein Underrun-Fehler ein.
DWORD dw_OnBoardBufferSize	Größe (= n von 1024 * 2 ⁿ Bytes) des Speichers auf der Karte (im RAM auf der Karte), der für das analoge Ausgangsmodul verwendet werden soll.
DWORD dw_CompareValue	Definiert den Platz, der auf dem RAM der Karte mindestens frei sein muss, um einen Interrupt zu erzeugen.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert den Signalgenerator für den analogen Ausgangskanal *b_Channel*.
Lädt den RAM-Speicher auf der Karte mit den Werten des PC-Speichers.

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

```
i_ReturnValue = i_PCI3600_InitAnalogOutputSignalGenerator  
                (dw_BoardHandle,  
                 0,  
                 PCI3600_FREE_MODE,  
                 0,  
                 15);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Falscher Parameter-Mode

-4: Der Kanal ist nicht initialisiert

-5: Der Speicher ist nicht reserviert

-6: Der Compare-Value kann nicht größer als die Größe des Speichers auf der Karte sein und er kann nicht Null betragen.

-7: Der Signalgenerator ist schon von einem anderen Prozess initialisiert.

-8: Der Signalgenerator läuft

-9: Nicht genügend Platz im RAM auf der Karte

-10: Eine zyklische Erfassung oder eine Signalgenerator läuft.

-11: Der Compare-Value muss größer sein als der tatsächliche freie Platz im RAM auf der Karte.

-12: Die Größe des PC-RAM-Speichers muss mit der Größe des Speichers auf der Karte identisch sein und voll sein.

-13: Die Interruptroutine ist nicht installiert

-14: Zeitfunktionsfehler

-15: Time out tritt ein

-16: Bereitet den DMA-Transferfehler vor

35) i_PCI3600_StartAnalogOutputSignalGenerator()**Syntax:**

`_INT_ i_PCI3600_StartAnalogOutputSignalGenerator(DWORD
dw_BoardHandle,
BYTE b_Channel)`

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Startet den Signalgenerator des analogen Ausgangskanals *b_Channel*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_StartAnalogOutputSignalGenerator  
(dw_BoardHandle,  
0);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Signalgenerator ist nicht initialisiert oder nicht von diesem Prozess initialisiert

-4: Der Signalgenerator ist schon gestartet

36) i_PCI3600_GetAnalogOutputSignalGeneratorStatus()**Syntax:**

```

_INT_ i_PCI3600_GetAnalogOutputSignalGeneratorStatus
        (DWORD dw_BoardHandle,
         BYTE   b_Channel,
         PBYTE  pb_Status,
         PDWORD

pdw_NbrOfValueInOnBoardRAM,
                                     PDWORD

pdw_NbrOfValueInComputerRAM)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

PBYTE pb_Status	Status des Signalgenerators 0: nicht gestartet 1: gestartet
PDWORD pdw_NbrOfValueInOnBoardRAM	Anzahl der Werte im RAM-Speicher auf der Karte.
PDWORD pdw_NbrOfValueInComputerRAM	Anzahl der Werte im PC-RAM-Speicher..

Aufgabe:

Gibt den Status und die Anzahl der Werte im RAM auf der Karte und im PC-RAM für den analogen Ausgangskanal *b_Channel* zurück.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT    i_ReturnValue;
BYTE   b_Status;
DWORD  dw_NbrOfValueOnBoardRAM;
DWORD  dw_NbrOfValueComputerRAM;

i_ReturnValue = i_PCI3600_GetAnalogOutputSignalGeneratorStatus
        (dw_BoardHandle,
         0,
         &b_Status,

&dw_NbrOfValueOnBoardRAM,

&dw_NbrOfValueComputerRAM);

```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Signalgenerator ist nicht gestartet oder wurde von einem anderen Prozess gestartet.

37) i_PCI3600_StopAnalogOutputSignalGenerator()**Syntax:**

`_INT_ i_PCI3600_StopAnalogOutputSignalGenerator(DWORD
dw_BoardHandle,
BYTE b_Channel)`

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte
BYTE b_Channel Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Stoppt den Signalgenerator des analogen Ausgangskanals *b_Channel*.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue =
i_PCI3600_StopAnalogOutputSignalGenerator(dw_BoardHandle,
0);

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Kanalnummer
-3: Der Signalgenerator ist nicht gestartet oder wurde von einem anderen Prozess
gestartet.

38) i_PCI3600_ReleaseAnalogOutputSignalGenerator()**Syntax:**

`_INT_ i_PCI3600_ReleaseAnalogOutputSignalGenerator(DWORD
dw_BoardHandle,
BYTE b_Channel)`

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt den Signalgenerator des analogen Ausgangskanals *b_Channel* frei.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

`i_ReturnValue = i_PCI3600_ReleaseAnalogOutputSignalGenerator
(dw_BoardHandle,
0);`

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Kanalnummer

-3: Der Signalgenerator ist nicht initialisiert oder ist nicht von diesem Prozess initialisiert.

39) i_PCI3600_AnalogOutput_EnableDisableDigitalInputExternalTrigger()**Syntax:**

```

_INT_ i_PCI3600_AnalogOutput_EnableDisableDigitalInputExternalTrigger
                                     (DWORD
dw_BoardHandle,
                                     BYTE
b_Channel,
                                     BYTE
b_ExternTriggerFlag)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Index des analogen Ausgangskanals (0 bis 1)
BYTE b_ExternTriggerFlag	Aktiviert/deaktiviert das Flag für den externen Trigger des digitalen Eingangs. 0 : Deaktivieren 1 : Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den externen Trigger des digitalen Eingangs (digitaler Eingang 0) für den analogen Ausgangskanal *b_Channel*.

- Ausgabe

Es erfolgt keine Ausgabe

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT     i_ReturnValue;

```

```

i_ReturnValue =
i_PCI3600_AnalogOutput_EnableDisableDigitalInputExternalTrigger
(dw_BoardHandle,
0,
PCI3600_ENABLE);

```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Kanalnummer
-3: Flag des extern Triggers ist falsch
-4: Der Kanal ist nicht initialisiert
-5: Der digitale externe Trigger ist nicht initialisiert

9.1.4 Chronometer-Modul

40) i_PCI3600_InitChronometerModule()

Syntax:

```
_INT_ i_PCI3600_InitChronometerModule(DWORD dw_BoardHandle,
                                         BYTE    b_ChronometerModule,
                                         BYTE    b_ClockDivisor)
```

Parameter:

- Eingabe:

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)
BYTE b_InputDivisor	Eingangs-Divisor (0 bis 15)
	0 : $2^0 \rightarrow \text{divisor} = 1$
	1 : $2^1 \rightarrow \text{divisor} = 2$
	...

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert das Chronometer-Modul *b_ChronometerModule*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;
INT     i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_InitChronometerModule(dw_BoardHandle,
                                                  0,
                                                  1);
```

Return-Wert:

- 0 : Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2: Falsche Chronometer-Nummer
- 3: Falsche Auswahl des Clock-Divisors
- 4: Der Chronometer ist schon von einem anderen Prozess initialisiert
- 5: Initialisierungsfehler

41) i_PCI3600_ReleaseChronometerModule()**Syntax:**

INT i_PCI3600_ReleaseChronometerModule (DWORD dw_BoardHandle,
BYTE b_ChronometerModule)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte
BYTE b_ChronometerModule
 Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt das Chronometer-Modul *b_ChronometerModule* frei.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_ReleaseChronometerModule (dw_BoardHandle,
0);

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Chronometer-Nummer

-3: Der Chronometer ist nicht initialisiert oder ist von einem anderen Prozess
initialisiert

42) i_PCI3600_StartChronometerModuleAcquisition()**Syntax:**

```
_INT_ i_PCI3600_StartChronometerModuleAcquisition(DWORD  
dw_BoardHandle,  
  
BYTE  
b_ChronometerModule)
```

Parameter:**-Eingabe:**

DWORD	dw_BoardHandle	Handle der Karte
BYTE	b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Startet die Erfassung des Chronometer-Moduls.

Wenn diese Funktion verwendet wird, wird der Wert des Chronometers gelatcht und im internen FIFO der Karte gespeichert

Mit folgender Funktion können Sie die Erfassung stoppen:

i_PCI3600_StopChronometerModuleAcquisition().

Die Erfassung wird automatisch gestoppt, sobald der FIFO voll ist.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_StartChronometerModuleAcquisition  
(dw_BoardHandle,  
  
0);
```

Return-Wert:

- 0 : Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2: Falsche Chronometer-Nummer
- 3: Der Chronometer ist nicht initialisiert
- 4: Die Erfassung wurde schon gestartet

43) i_PCI3600_GetChronometerModuleFIFOStatus()**Syntax:**

```

_INT_ i_PCI3600_GetChronometerModuleFIFOStatus(DWORD
dw_BoardHandle,
                                                    BYTE
b_ChronometerModule,
                                                    PBYTE
pb_FIFOStatus)

```

Parameter:**-Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

PBYTE pb_FIFOStatus	interner FIFO-Status
	00 : FIFO leer
	01 : FIFO nicht leer
	10 : FIFO-Fehler (Overflow)

Aufgabe:

Gibt den Status des internen FIFOs zurück, um herauszufinden, ob die Chronometer-Werte verfügbar sind.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT i_ReturnValue;
BYTE b_FIFOStatus;

```

```

i_ReturnValue = i_PCI3600_GetChronometerModuleFIFOStatus
(dw_BoardHandle,

```

```

0,

```

```

&b_FIFOStatus);

```

Return-Wert:

```

0: Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Chronometer-Nummer
-3: Der Chronometer ist nicht initialisiert

```

44) i_PCI3600_ReadChronometerModuleValue()**Syntax:**

```
_INT_ i_PCI3600_ReadChronometerModuleValue  
                                (DWORD dw_BoardHandle,  
                                BYTE     b_ChronometerModule,  
                                PDWORD pdw_ChronometerValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte
BYTE b_ChronometerModule
 Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

PDWORD pdw_ChronometerValue
 Chronometer-Wert, gelesen vom internen
 FIFO

Aufgabe:

Liest einen Chronometer-Wert vom internen FIFO.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT        i_ReturnValue;  
DWORD   dw_ChronometerValue;
```

```
i_ReturnValue = i_PCI3600_ReadChronometerModuleValue (dw_BoardHandle,  
                                                         0,
```

```
&dw_ChronometerValue);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Chronometer-Nummer
-3: Der Chronometer ist nicht initialisiert
-4: FIFO ist leer

45) i_PCI3600_StopChronometerModuleAcquisition()**Syntax:**

INT i_PCI3600_StopChronometerModuleAcquisition(DWORD
dw_BoardHandle,

BYTE
b_ChronometerModule)

Parameter:**-Eingabe:**

DWORD dw_BoardHandle Handle der Karte
BYTE b_ChronometerModule
 Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Stoppt die Erfassung des Wertes des Chronometer-Moduls.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_StopChronometerModuleAcquisition
(dw_BoardHandle,

0);

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Falsche Chronometer-Nummer
-3: Der Chronometer ist nicht initialisiert oder ist von einem anderen Prozess
initialisiert

46) i_PCI3600_InitChronometerModuleCyclicAcquisition**Syntax:**

```

_INT_ i_PCI3600_InitChronometerModuleCyclicAcquisition
        (DWORD dw_BoardHandle,
         BYTE b_ChronometerModule,
         BYTE b_Mode,
         DWORD
dw_OnBoardBufferSize,
         DWORD dw_CompareValue)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)
BYTE b_Mode	Definiert den verwendeten Mode
	0: AUTO STOP-Mode
	In diesem Mode ist der Speicher auf der Karte mit Werten gefüllt.
	Wenn der Speicher auf der Karte voll ist, wird die Erfassung gestoppt und ein Interrupt wird ausgelöst.
	1: RING Buffer Mode
	In diesem Mode befinden sich immer Werte im Speicher auf der Karte.
	In beiden Modes:
	Jedesmal, wenn die Werte im Speicher dw_CompareValue erreichen, wird ein Interrupt ausgelöst, so dass die Software den Wert vom Speicher ablesen kann.
DWORD dw_OnBoardBufferSize	Größe (in nbr Bytes : $1024 * 2^{dw\_OnBoardBufferSize}$) des Speichers auf der Karte (im RAM auf der Karte), die für den Chrono. verwendet wird.
DWORD dw_CompareValue	Definiert die Anzahl von Erfassungen, die im Speicher auf der Karte stattfinden müssen, um einen Interrupt zum Starten eines DMA-Transfers zu generieren.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert die zyklische Erfassung für das Chronometer-Modul *b_ChronometerModule*.

Funktionsaufruf:

DWORD dw_BoardHandle;

```
INT      i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_InitChronometerModuleCyclicAcquisition  
                (DWORD dw_BoardHandle,  
                 0,  
                 PCI3600_AUTO_STOP_MODE,  
                 0,  
                 10);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Falsche Modul-Nummer

-3: Falscher Parameter-Mode

-4: Das Modul ist nicht initialisiert

-5: Kein PC-Speicher verfügbar

-6: Der Compare-Value kann Folgendes nicht sein:

- größer als die Speichergröße auf der Karte im Auto Stop-Mode

- größer als oder gleich als die Größe des Speichers auf der Karte im Ring
Buffer Mode

- Null

-7: Die zyklische Erfassung wurde schon von einem anderen Prozess initialisiert

-8: Die zyklische Erfassung läuft.

-9: Es ist nicht genügend Platz im RAM der Karte vorhanden

-10: Die Interruptroutine ist nicht installiert

-11: Init Chronometer-Fehler

47) i_PCI3600_StartChronometerModuleCyclicAcquisition()**Syntax:**

```
_INT_ i_PCI3600_StartChronometerModuleCyclicAcquisition  
                                     (DWORD dw_BoardHandle,  
                                     BYTE  
b_ChronometerModule)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte
BYTE b_ChronometerModule Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Startet die zyklische Erfassung für das Chronometer-Modul
b_ChronometerModule.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT        i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_StartChronometerModuleCyclicAcquisition  
(dw_BoardHandle,  
                                     0);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2 : Falsche Modulnummer
-3: Die zyklische Erfassung ist nicht initialisiert oder wurde nicht von diesem
Prozess initialisiert
-4: Die zyklische Erfassung wurde schon gestartet

48) i_PCI3600_GetChronometerModuleCyclicAcquisitionStatus()

Syntax:

```

_INT_ i_PCI3600_GetChronometerModuleCyclicAcquisitionStatus
                                     (DWORD
dw_BoardHandle,
                                     BYTE
b_ChronometerModule,
                                     PBYTE pb_Status,
                                     PDWORD
pdw_NbrOfValue)

```

Parameter:

-Eingabe

DWORD dw_BoardHandle Handle der Karte
 BYTE b_ChronometerModule Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

PBYTE pb_Status Status der zyklischen Erfassung
 0: nicht gestartet
 1: gestartet
 PDWORD pdw_NbrOfValue Anzahl der Werte im RAM-Speicher auf der Karte für den Chronometer.

Aufgabe:

Gibt den Status und die Anzahl der Werte im RAM-Speicher für das Chronometer-Modul *b_ChronometerModule* zurück.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT        i_ReturnValue;
BYTE b_Status;
DWORD dw_NbrOfValue;

i_ReturnValue = i_PCI3600_GetChronometerModuleCyclicAcquisitionStatus

(dw_BoardHandle,
                                     0,
                                     &b_Status,

&dw_NbrOfValue);

```

Return-Wert:

0 : Kein Fehler
 -1: Der Handle-Parameter der Karte ist falsch
 -2: Falsche Modulnummer
 -3: Die zyklische Erfassung wurde nicht gestartet oder wurde von einem anderen Prozess gestartet.

49) i_PCI3600_IncrementChronometerCyclicAcquisitionBufferReadPointer()**Syntax:**

```
_INT_ i_PCI3600_IncrementChronometerCyclicAcquisitionBufferReadPointer  
      (DWORD dw_BoardHandle,  
       DWORD dw_NbrOfValue)
```

Parameter:**- Eingabe**

DWORD dw_BoardHandle	Handle der Karte
DWORD dw_NbrOfValue	Anzahl der Werte

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Ermöglicht die Inkrementierung des Lesezeigers des Ring Buffers, der für den DMA-Transfer verwendet wird.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT      i_ReturnValue;  
DWORD dw_BufferHandle;
```

```
i_ReturnValue =  
i_PCI3600_IncrementChronometerCyclicAcquisitionBufferReadPointer  
  
(dw_BoardHandle,  
  
  
1000);
```

Return-Wert:

0 : Kein Fehler
-1 : Der Handle-Parameter der Karte ist falsch

50) i_PCI3600_StopChronometerModuleCyclicAcquisition()**Syntax:**

```
_INT_ i_PCI3600_StopChronometerModuleContinuousAcquisition  
                                     (DWORD  
dw_BoardHandle,  
                                     BYTE  
b_ChronometerModule)
```

Parameter:**- Eingabe**

DWORD dw_BoardHandle Handle der Karte
BYTE b_ChronometerModule Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Stoppt die zyklische Erfassung für das Chronometer-Modul
b_ChronometerModule.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT      i_ReturnValue;
```

```
i_ReturnValue =  
i_PCI3600_StopChronometerModuleContinuousAcquisition(dw_BoardHandle,  
0);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2 : Falsche Modulnummer
-3: Die zyklische Erfassung ist nicht gestartet oder wurde von einem anderen
Prozess gestartet.

51) i_PCI3600_ReleaseChronometerModuleCyclicAcquisition()**Syntax:**

```
_INT_ i_PCI3600_ReleaseChronometerModuleCyclicAcquisition  
                                     (DWORD dw_BoardHandle,  
                                     BYTE  
b_ChronometerModule)
```

Parameter:**- Eingabe**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Gibt die zyklische Erfassung für das Chronometer-Modul *b_ChronometerModule* frei.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_ReleaseChronometerModuleCyclicAcquisition
```

```
(dw_BoardHandle,  
                                     0);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2 : Falsche Modulnummer

-3 : Die zyklische Erfassung ist nicht initialisiert oder wurde nicht von diesem Prozess initialisiert

52) i_PCI3600_Chronometer_EnableDisableDigitalInputExternalTrigger()**Syntax:**

```

_INT_ i_PCI3600_Chronometer_EnableDisableDigitalInputExternalTrigger
        (DWORD dw_BoardHandle,
         BYTE    b_ChronometerModule,
         BYTE    b_ExternTriggerFlag)

```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometer-Moduls (0 bis 3)
BYTE b_ExternTriggerFlag	Aktiviert/deaktiviert das Flag für den externen Trigger der digitalen Eingänge
	0 : Deaktivieren
	1 : Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den externen Trigger der digitalen Eingänge (digitaler Eingang 0) für das Chronometer-Modul *b_ChronometerModule*.

Funktionsaufruf:

```

DWORD dw_BoardHandle;
INT    i_ReturnValue;

```

```

i_ReturnValue =
i_PCI3600_Chronometer_EnableDisableDigitalInputExternalTrigger
        (dw_BoardHandle,
         0,
         PCI3600_ENABLE);

```

Return-Wert:

- 0 : Kein Fehler
- 1: Der Handle-Parameter der Karte ist falsch
- 2 : Falsche Chronometernummer
- 3: Flag des externen Triggers ist falsch
- 4: Der Chronometer ist nicht initialisiert
- 5: Der digitale externe Trigger ist nicht initialisiert

9.1.5 Digitale Eingänge

53) i_PCI3600_Read1DigitalInput()

Syntax:

```
_INT_ i_PCI3600_Read1DigitalInput(DWORD dw_BoardHandle,  
                                   BYTE b_Channel,  
                                   PBYTE pb_ChannelValue)
```

Parameter:**-Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Zu lesender Kanal (0-7)

- Ausgabe

PBYTE pb_ChannelValue	Kanalwert :
	0 : Niedrig
	1 : Hoch

Aufgabe:

Gibt den Status eines Eingangskanals an. Die Variable *b_Channel* passiert den Eingangskanal, der zu lesen ist (0 bis 7). Mit der Variablen *pb_ChannelValue* wird ein Wert zurückgegeben 0 (niedrig) oder 1 (hoch).

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;  
BYTE b_ChannelValue;
```

```
i_ReturnValue = i_PCI3600_Read1DigitalInput (dw_BoardHandle,  
                                              0,  
                                              &b_ChannelValue);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Der Parameter der Kanalnummer ist falsch

54) i_PCI3600_Read4DigitalInputs()**Syntax:**

```
_INT_ i_PCI3600_Read4DigitalInputs(DWORD dw_BoardHandle,  
                                     BYTE b_Port,  
                                     PBYTE pb_PortValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Port	Nummer des Eingangports, den Sie lesen möchten (0 oder 1)

- Ausgabe

PBYTE pb_PortValue	Status des digitalen Eingangports (0 bis 15)
--------------------	--

Aufgabe:

Gibt den Status des Ports an. Die Variable *b_Port* passiert den Port, der zu lesen ist (1 oder 2). Mit der Variablen *pb_PortValue* wird ein Wert zurückgegeben

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;  
BYTE b_PortValue;
```

```
i_ReturnValue = i_PCI3600_Read4DigitalInputs (dw_BoardHandle,  
                                              0,  
                                              &b_PortValue);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Der Parameter der Kanalnummer ist falsch

55) i_PCI3600_Read8DigitalInputs0

Syntax:

```

_INT_i_PCI3600_Read8DigitalInputs(DWORD dw_BoardHandle,
                                   PBYTE pb_Value)

```

Parameter:

- Eingabe:

DWORD dw_BoardHandle	Handle der Karte
----------------------	------------------

- Ausgabe:

PBYTE	pb_Value	Status der digitalen Eingänge (0 bis 255)
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	23
24	24	24
25	25	25
26	26	26
27	27	27
28	28	28
29	29	29
30	30	30
31	31	31
32	32	32
33	33	33
34	34	34
35	35	35
36	36	36
37	37	37
38	38	38
39	39	39
40	40	40
41	41	41
42	42	42
43	43	43
44	44	44
45	45	45
46	46	46
47	47	47
48	48	48
49	49	49
50	50	50
51	51	51
52	52	52
53	53	53
54	54	54
55	55	55
56	56	56
57	57	57
58	58	58
59	59	59
60	60	60
61	61	61
62	62	62
63	63	63
64	64	64
65	65	65
66	66	66
67	67	67
68	68	68
69	69	69
70	70	70
71	71	71
72	72	72
73	73	73
74	74	74
75	75	75
76	76	76
77	77	77
78	78	78
79	79	79
80	80	80
81	81	81
82	82	82
83	83	83
84	84	84
85	85	85
86	86	86
87	87	87
88	88	88
89	89	89
90	90	90
91	91	91
92	92	92
93	93	93
94	94	94
95	95	95
96	96	96
97	97	97
98	98	98
99	99	99
100	100	100
101	101	101
102	102	102
103	103	103
104	104	104
105	105	105
106	106	106
107	107	107
108	108	108
109	109	109
110	110	110
111	111	111
112	112	112
113	113	113
114	114	114
115	115	115
116	116	116
117	117	117
118	118	118
119	119	119
120	120	120
121	121	121
122	122	122
123	123	123
124	124	124
125	125	125
126	126	126
127	127	127
128	128	

Aufgabe:

Gibt den Status der digitalen Eingänge an.

Funktionsaufruf:

```
DWORD dw_BoardHandle;
```

```
INT i ReturnValue;
```

BYTE b_ Value;

```
i_ReturnValue = i_PCI3600_Read8DigitalInputs(dw_BoardHandle,
                                              &b_Value);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

9.1.6 Digitale Ausgänge

56) i_PCI3600_EnableDisableDigitalOutputMemory()

Syntax:

INT i_PCI3600_EnableDisableDigitalOutputMemory (DWORD
dw_BoardHandle,
BYTE b_Enable)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Enable	0: deaktiviert den digitalen Ausgangsspeicher 1 : aktiviert den digitalen Ausgangsspeicher

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den digitalen Ausgangsspeicher.

Nach dem Aufruf zur Aktivierung des digitalen Ausgangsspeichers, werden die Ausgangskanäle, die sie zuvor mit der Funktion "i_PCI3600_SetXDigitalOutputOn" aktiviert haben, nicht zurückgesetzt. Sie können Sie mit der Funktion "i_PCI3600_SetXDigitalOutputOff" zurücksetzen.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_EnableDisableDigitalOutputMemory
(dw_BoardHandle,

PCI3600_ENABLE);

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch.

57) i_PCI3600_Set1DigitalOutputOn()**Syntax:**

INT i_PCI3600_Set1DigitalOutputOn (DWORD dw_BoardHandle,
BYTE b_Channel)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Anzahl der Ausgangskanäle, die Sie setzen möchten (0 bis 7).

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt die Ausgangskanäle, die durch *b_Channel* passiert sind.

Das Setzen eines Ausgangskanals bedeutet das Setzen eines Ausgangskanals auf „hoch“.

Anschalten des digitalen Ausgangsspeichers (ON)

Siehe Funktion "i_PCI3600_EnableDisableDigitalOutputMemory (...)"

b_Channel= 1

Der Ausgangskanal1 ist gesetzt. Die anderen Ausgangskanäle behalten ihren Status.

Ausschalten des digitalen Ausgangsspeichers (OFF)

Siehe Funktion "i_PCI3600_EnableDisableDigitalOutputMemory (...)"

b_Channel= 1

Der Ausgangskanal 1 ist gesetzt. Die anderen Ausgangskanäle sind zurückgesetzt. Falls Sie den digitalen Ausgangsspeichers ausgeschaltet haben (OFF), werden alle anderen Kanäle auf "0" gesetzt.

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

i_ReturnValue = i_PCI3600_Set1DigitalOutputOn(dw_BoardHandle,
0);

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2 : Die Kanalanzahl liegt nicht zwischen 0 und 7.

58) i_PCI3600_Set1DigitalOutputOff()**Syntax:**

```
_INT_ i_PCI3600_Set1DigitalOutputOff(DWORD dw_BoardHandle,  
                                     BYTE b_Channel)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Channel	Anzahl der Ausgangskanäle, die Sie zurücksetzen möchten (0 bis 7).

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt den Ausgangskanal, den Sie durch *b_Channel* passiert haben, zurück. Das Zurücksetzen eines Ausgangskanals bedeutet das Setzen auf „niedrig“.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_Set1DigitalOutputOff(dw_BoardHandle,  
                                              0);
```

Return-Wert:

0 : Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2 : Falsche Kanalnummer

-3: Der digitale Ausgangsspeicher ist nicht angeschaltet.

59) i_PCI3600_Set4DigitalOutputsOn()**Syntax:**

```
_INT_ i_PCI3600_Set4DigitalOutputsOn(DWORD dw_BoardHandle,  
                                     BYTE b_Port,  
                                     BYTE b_PortValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Port	Nummer des Ausgangsports (0 oder 1)
BYTE b_PortValue	Ausgangswert (0 bis 15)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt einen oder mehrere Ausgangskanäle eines Ports. Das Setzen eines Ausgangskanals bedeutet das Setzen auf „hoch“.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_Set4DigitalOutputsOn(dw_BoardHandle,  
                                                0,  
                                                15);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Die Portnummer ist falsch
-3: Der Portwert ist falsch

60) i_PCI3600_Set4DigitalOutputsOff()**Syntax:**

```
_INT_ i_PCI3600_Set4DigitalOutputsOff(DWORD dw_BoardHandle,  
                                       BYTE b_Port,  
                                       BYTE b_PortValue)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Port	Nummer des Ausgangports (0 oder 1)
BYTE b_PortValue	Ausgangswert (0 bis 15)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt ein oder mehrere Ausgangskanäle eines Ports zurück. Zurücksetzen bedeutet das Setzen auf „niedrig“.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_Set4DigitalOutputsOff(dw_BoardHandle,  
                                                0,  
                                                15);
```

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch
-2: Digitaler Ausgangsspeicher ist nicht angeschaltet.
-3: Die Portnummer ist falsch.
-4: Der Portwert ist falsch.

61) i_PCI3600_Set8DigitalOutputsOn()**Syntax:**

INT i_PCI3600_Set8DigitalOutputsOn(DWORD dw_BoardHandle,
BYTE b_Value)

Parameter:**-Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Value	Ausgangswert (0 bis 255)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt ein oder mehrere Ausgangskanäle der Karte **APCI-3600**.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_Set8DigitalOutputsOn (dw_BoardHandle,
255);

Return-Wert:

0 : Kein Fehler
-1: Der Handle-Parameter der Karte ist falsch

62) i_PCI3600_Set8DigitalOutputsOff()**Syntax:**

INT i_PCI3600_Set8DigitalOutputsOff(DWORD dw_BoardHandle,
BYTE b_Value)

Parameter:**-Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Value	Ausgabewert (0 bis 255)

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Setzt ein oder mehrere Ausgangskanäle der Karte **APCI-3600** zurück.

Funktionsaufruf:

DWORD dw_BoardHandle;
INT i_ReturnValue;

i_ReturnValue = i_PCI3600_Set8DigitalOutputsOff(dw_BoardHandle,
15);

Return-Wert:

0: Kein Fehler

-1: Der Handle-Parameter der Karte ist falsch

-2: Der digitale Ausgangsspeicher ist nicht angeschaltet.

9.1.7 Externer Trigger

63) i_PCI3600_InitDigitalInputExternalTrigger0

Syntax:

```
_INT_ i_PCI3600_InitDigitalInputExternalTrigger(DWORD dw_BoardHandle,
                                                BYTE    b_TriggerLevel,
                                                WORD
w_TriggerCountValue)
```

Parameter:

- Eingabe:

DWORD dw_BoardHandle	Handle der Karte
BYTE b_TriggerLevel	Definiert die Vorderseite des digitalen Eingangs, der den externen Trigger aktiviert 01: steigende Flanke 10: fallende Flanke 11: beide Flanken
WORD w_TriggerCountValue	Definiert die Anzahl der Trigger, die darauf warten sollen, dass die Firmware den Trigger erkennt.

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Initialisiert den externen Trigger des digitalen Eingangs (digitaler Eingang 0).

Funktionsaufruf:

```
DWORD dw_BoardHandle;
INT     i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_InitDigitalInputExternalTrigger (dw_BoardHandle,
                                                            1,
                                                            1);
```

Return-Wert:

0 : Kein Fehler
 -1: Der Handle-Parameter der Karte ist falsch
 -2 : Falsche Auswahl der Triggerebene
 -3 : Falscher Zählwert des Triggers
 -4: Der externe Trigger wurde schon durch einen anderen Prozess initialisiert.

9.1.8 Mastertrigger

64) _INT_ i_PCI3600_AnalogInput_EnableDisableMasterTrigger 0 :

Syntax:

```
_INT_ i_PCI3600_AnalogInput_EnableDisableMasterTrigger
      (DWORD dw_BoardHandle,
       BYTE   b_Module,
       BYTE   b_MasterTriggerFlag)
```

Parameter:

-Eingabe:

DWORD dw_BoardHandle	Handle der Karte
BYTE b_Module	Index des analogen Eingangmoduls (0 bis 3)
BYTE b_MasterTriggerFlag	Aktiviert/deaktiviert das Flag für den Mastertrigger 0 : Deaktivieren 1 : Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den Mastertrigger für das analoge Eingangsmodul *b_Module*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;
INT     i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_AnalogInput_EnableDisableMasterTrigger
                (dw_BoardHandle,
                 0,
                 PCI3600_ENABLE);
```

Rückgabewert:

0: Kein Fehler
 -1: Handleparameter der Karte ist falsch
 -2: Falsche Modulnummer
 -3: Externes Triggerflag ist falsch
 -4: Das Modul ist nicht initialisiert

65) _INT_ i_PCI3600_AnalogOutput_EnableDisableMasterTrigger 0 :**Syntax:**

```
_INT_ i_PCI3600_AnalogOutput_EnableDisableMasterTrigger  
                                (DWORD    dw_BoardHandle,  
                                BYTE      b_Channel,  
                                BYTE      b_MasterTriggerFlag)
```

Parameter:**- Eingabe:**

DWORD	dw_BoardHandle	Handle der Karte
BYTE	b_Channel	Index des analogen Ausgangskanals (0 bis 1)
BYTE	b_MasterTriggerFlag	Aktivieren/deaktivieren des Flags für den Mastertrigger 0: Deaktivieren 1: Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den Mastertrigger für den analogen Ausgangskanal *b_Channel*.

- Ausgabe

Es erfolgt keine Ausgabe

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT    i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_AnalogOutput_EnableDisableMasterTrigger  
                (dw_BoardHandle,  
                0,  
                PCI3600_ENABLE);
```

Rückgabewert:

0: Kein Fehler
-1: Handleparameter der Karte ist falsch
-2: Falsche Kanalnummer
-3: Externe Triggerflag ist falsch
-4: Der Kanal ist nicht initialisiert

66) _INT_i_PCI3600_Chronometer_EnableDisableMasterTrigger 0 :**Syntax:**

```
_INT_i_PCI3600_Chronometer_EnableDisableMasterTrigger  
        (DWORD dw_BoardHandle,  
         BYTE    b_ChronometerModule,  
         BYTE    b_MasterTriggerFlag)
```

Parameter:**- Eingabe:**

DWORD dw_BoardHandle	Handle der Karte
BYTE b_ChronometerModule	Index des Chronometermoduls (0 bis 3)
BYTE b_MasterTriggerFlag	Aktiviert/deaktiviert das Flag für den Mastertrigger 0: Deaktivieren 1: Aktivieren

- Ausgabe

Es erfolgt keine Ausgabe

Aufgabe:

Aktiviert/deaktiviert den Mastertrigger für das Chronometermodul *b_ChronometerModule*.

Funktionsaufruf:

```
DWORD dw_BoardHandle;  
INT i_ReturnValue;
```

```
i_ReturnValue = i_PCI3600_Chronometer_EnableDisableMasterTrigger  
                (dw_BoardHandle,  
                 0,  
                 PCI3600_ENABLE);
```

Rückgabewert:

0: Kein Fehler
-1: Handleparameter der Karte ist falsch
-2: Falsche Chronometernummer
-3: Externes Triggerflag ist falsch
-4: Der Chronometer ist nicht initialisiert

67) _INT_ i_PCI3600_MasterTrigger 0 :**Syntax:**

INT i_PCI3600_MasterTrigger (DWORD dw_BoardHandle)

Parameter:**- Eingabe:**

DWORD dw_BoardHandle Handle der Karte

- Ausgabe:

Es erfolgt keine Ausgabe

Aufgabe:

Generiert einen Mastertrigger. Kann nur verwendet werden, wenn die Karte im Mastermode initialisiert ist.

Funktionsaufruf:

DWORD dw_BoardHandle;

INT i_ReturnValue;

i_ReturnValue = i_PCI3600_MasterTrigger (dw_BoardHandle);

Rückgabewert:

0: Kein Fehler

-1: Handleparameter der Karte ist falsch

-2: Die Karte ist nicht als Master initialisiert

10 ANHANG

10.1 Verwendete Abkürzungen

Abkürzung	
ADC	A/D converter (= A/D-Wandler)
Clk	Clock
CNT	Counter
DIFF	Differentiell
DMA	Direct Memory Access
DUT	Device Under Test
EEPROM	Electrically erasable PROM (elektrisch löschbarer PROM)
FFT	Fast Fourier Transformation
FIFO	First In First Out
Gen	Generator
ICP	Integrated Circuit Piezoelectric
IRQ	Interrupt
L	Linker Kanal (eines A/D-Wandlers)
LRCK	Abtastfrequenz
M1	Mode-Signal 1
MCLK	Clock Signal
PCI	Peripheral Component Interconnect
PLD	Programmable Logic Device
R	Rechter Kanal (eines A/D-Wandlers)
RAM	Random Access Memory
RMS	Root Mean Square
RS485	Schnittstelle
SCLK	Clock Signal
SDRAM	Synchro Dynamic Random Access Memory
SE	Single- Ended
SINAD = S/N + D	Signal-to-Noise-and-Distortion Ratio

10.2 Glossar

Begriff	Erklärung
A/D-Wandler (ADC)	= <i>ADC</i> Ein A/D-Wandler wandelt Analogsignale in digitale Werte um.
Abtastfrequenz	= <i>Abtastrate</i> Die Abtastfrequenz gibt an, in welcher Feinheit ein analoges Signal diskretisiert, also portioniert wird. Zur Unterteilung eines analogen Signals werden in (meist) regelmäßigen Abständen Proben (Samples) genommen und in numerische Werte umgewandelt. Das Verhältnis von Anzahl der Samples (Proben) zur zeitlichen oder räumlichen Einheit ist die Abtastrate.
Abtastung	Verfahren, in dem durch A/D-Wandlung aus analogem Audiomaterial Werte abgetastet und digital gespeichert werden.
ADC	= <i>A/D-Wandler</i>
Aliasingfrequenz	= Aliasing-Effekt Beim Digitalisieren können Frequenzen entstehen, die im Original nicht vorhanden sind. Diese werden „Aliasingfrequenzen“ genannt und sind das Ergebnis einer unzureichenden Abtastfrequenz. Wird z.B. ein Signal mit der Abtastfrequenz $2f(t)$ abgetastet, das höhere Frequenzen als $f(t)$ enthält, entstehen Frequenzen, die im Original überhaupt nicht vorhanden sind, d.h. durch eine zu geringe Abtastrate gehen nicht nur Frequenzen des Originals verloren, sondern es kommen auch neue – falsche – hinzu. Um Aliasing-Fehler zu vermeiden, schaltet man dem analogen Eingang einen Anti-Aliasing-Filter vor. (Siehe auch „Nyquist-Konzept“ und „Anti-Aliasing-Filter“)
Anti-Aliasing-Filter	Analoge Vorfilter, die Aliasing-Effekte verhindern. Hierbei werden vor dem Abtasten Frequenzen, die größer sind als die halbe Abtastfrequenz herausgefiltert. (Siehe auch „Aliasingfrequenz“ und „Abtastfrequenz“)
Auflösung	Die kleinste Änderung, die von einem A/D-Wandler erkannt oder von einem D/A-Wandler produziert werden kann.
Bandbreite	= <i>bandwidth</i> Die Frequenz, bei der sich die Verstärkung eines Verstärkers oder anderen Schaltkreises um 3 dB gegenüber dem Gleichspannungswert verringert hat. Gilt ebenso für jene Frequenzbereiche, innerhalb derer – ausgehend von einem mittleren Frequenzwert – die Abschwächung weniger als 3 dB beträgt.

Clock	Ein Schaltkreis, der zur Synchronisation des Wandlerbetriebes Zeitgabe- bzw. Taktimpulse erzeugt.
Crosstalk	= <i>Übersprechen</i> Gilt für Analogmultiplexer und kennzeichnet das Verhältnis von Ausgangsspannung zu Eingangsspannung, wobei alle Kanäle parallel und ausgeschaltet sind. Wird gewöhnlich als Eingangs- zu Ausgangs-Abschwächungsverhältnis bezeichnet und in dB ausgedrückt.
Differentielle Eingänge (DIFF)	<i>Zwei-Draht-Eingänge</i> Störsignale (die auf beide Leitungen wirken!) werden durch die Differenzbildung am Eingang nicht mit in die Messung einbezogen. Einsatz bei störungsbehafteten Messleitungen und größeren Leitungslängen.
DMA	= <i>Direct Memory Access</i> Direkter Speicherzugriff ohne Einsatz der CPU. Ermöglicht sehr schnellen Datentransfer innerhalb des Computers.
DUT	= <i>Device Under Test</i> Testobjekt (bei der Geräuschmessung z.B. ein Handy)
FFT	= <i>Fast Fourier-Transformation</i> Ein Algorithmus, der die Frequenzverteilung einer Kurvenform schnell bestimmen kann.
FIFO	= <i>First In First Out</i> Organisationsprinzip für die Bedienung von Warteschlangen, bei dem die Abarbeitung von Aufträgen in der gleichen Reihenfolge erfolgt wie die Annahme. So werden z.B. beim Leeren eines Speichers zuerst eingespeicherte Daten als erste wieder ausgegeben.
Gain	= <i>Verstärkung</i> Er dient zur Verstärkung oder Abschwächung eines analogen Signals. Er wirkt als Faktor auf ein Signal, z. B. ein Analogsignal, das dann auf einen A/D-Wandler geführt wird. Wird z.B. ein Eingangsbereich ± 5 V gewählt und die Verstärkung auf 10 gesetzt, so können Eingangssignale im $\pm 0,5$ V-Bereich gemessen werden.
Interrupt	= <i>Unterbrechung</i> Die Abarbeitung eines aktuellen Programms wird gestoppt bzw. unterbrochen und die CPU wird veranlasst, eine andere festgelegte Routine zu bearbeiten. Nach Abschluss dieser Routine wird in das unterbrochene Programm zurückgesprungen.
Klirrfaktor	= <i>THD</i> Wie entsteht der Klirrfaktor? Nichtlineare Verzerrungen können im Verstärker entstehen, da diese nicht exakt linear verstärken. Die Kurve ist keine echte Sinuskurve mehr, da sie Oberwellen enthält. Diese Oberwellen lassen sich mit Hilfe der Fouriertransformation errechnen. Der Klirrfaktor bleibt gering, wenn der Oberwellenanteil bzw. die nichtlineare Verzerrungen gering bleiben. (Siehe auch „FFT“)

Nyquist-Konzept	Laut dem Nyquist-Konzept muss die Abtastrate mindestens das Doppelte der zu digitalisierenden Frequenz betragen, damit aus dem so erhaltenen zeitdiskreten Signal das Originalsignal ohne Informationsverlust wieder rekonstruiert werden kann. D.h., dass ein kontinuierliches Signal mit einer Maximalfrequenz $f_{\max}$ mit einer Frequenz größer als $2 \cdot f_{\max}$ abgetastet werden muss. (Siehe auch: „Aliasingfrequenz“)
Optokuppler	Mit einem Optokuppler kann Gleichspannung übertragen werden. Der Vorteil liegt in der geringen Baugröße und den guten EMV-Eigenschaften
Oversampling	Das Signal wird mit einem Vielfachen der Abtastfrequenz abgetastet, so dass man keine zusätzlichen Informationen erhält. Doch der Aufwand für Verarbeitung und Speicherung ist jedoch beträchtlich.
PLD	= <i>Programmable Logic Device</i> Programmierbarer logischer Schaltkreis
Puffer	Pufferspeicher für den Ausgleich unterschiedlicher Arbeitsgeschwindigkeiten von Rechner und Peripheriegeräten
RMS	= <i>Root Mean Square</i> . Er ist der quadratische Mittel-/ Effektivwert Die Ausgangsspannung wird in RMS gemessen
RS485	Bei einer RS485-Schnittstelle handelt es sich genau genommen um einen Bus, da die Datenübertragung der einzelnen Teilnehmer nicht auf zwei begrenzt ist.
Sampler	Ein elektronischer Schalter, der zur Erzeugung einer Serie von analogen Abtastimpulsen mit hoher Geschwindigkeit ein- und ausgeschaltet wird.
Samplingfrequenz	= <i>Zeitliche Abtastrate</i> (siehe auch „Abtastrate“)
SINAD	= <i>Sound in Noise and Distortion</i>
Single Ended-Eingänge (SE)	Ein-Draht-Eingänge mit Bezug zur System-Masse. Störsignale gehen voll mit in die Messung ein. Einsatz bei relativ hohen Spannungspegeln und kurzen Leitungen
SNR	= <i>Signal to noise ratio</i> Tritt bei der Erfassung und Digitalisierung eines rauschfreien Eingangssignals (reiner Sinus) auf. Wird bei der Fast Fourier Transformation (FFT) in einem FFT-Plot als Verhältnis des Trägers zum Effektivwert aller Rauschanteile dargestellt und in dB angegeben. Das theoretische Signal/Rauschverhältnis für sinusförmige Signale beträgt: $SNR = 6.02n + 1.76$, wobei n = Auflösung des A/D-Wandlers in Bit ist.
Tiefpassfilter	Ein Tiefpassfilter ist ein Filter, der nur Frequenzen unterhalb der Grenzfrequenz passieren lässt
Verstärkungsfehler	= <i>Gain error</i> Die Differenz der Anstiegsteilheit der tatsächlichen und idealen Übertragungsfunktion bei einem Datenwandler oder anderen Baustein. Wird in Prozent der Analoggröße ausgedrückt.

11 INDEX

11.1

A

A/D-Wandler 48
 Abkürzungen 150
 Abtastfrequenz
 Analoge Ausgänge 54
 ADDIREG 27
 Schaltflächen 30
 Textfelder 29
 ADDIREG-Hauptfenster 28
 Aliasing-Effekt 44
 Allgemeine Beschreibung der Karte 10
 Allgemeine Funktionen
 Softwarefunktionen 69
 Amplituden-Fehler 20
 Analoge Ausgänge
 Signalgeneratormode (Free Run) 53
 Signalgeneratormode (Ring Buffer) 54
 Simple Mode 53
 Softwarfunktionen 107
 Analoge Eingänge
 Eingangsschaltung 42
 Funktionsbeschreibung 42
 Simple Mode 51
 Softwarefunktionen 90
 Verteilung 42
 Zyklischer Mode
 Auto Stop 52
 Analogfilter 47
 Anschluss an die Peripherie 37
 Anti-Aliasing Hintergrundinfos
 Analogfilter 47
 Anti-Aliasing-Filter 46
 Diskrete Abtastung von Analogsignalen 43
 Nyquist-Konzept 44
 Anti-Aliasing-Filter 46

B

Benutzer
 Persönliche Schutzausrüstung 12
 Qualifikation 12
 Bestimmungsgemäßer Zweck 10
 Bestimmungswidriger Zweck 10
 Bestückungsplan 23
 Betriebsspannung 16
 Betriebssystem 16
 Blockschaltbild 41
 Bus Geschwindigkeit 16

C

Chronometer-Modes 57
 Simple Mode 57
 Zyklischer Mode
 Auto Stop 57
 Ring Buffer 57
 Chronometer-Modul
 Softwarefunktionen 125
 Clock Generator 2 61
 Clock-Generator 1 60
 Clock-Generator Werte 60

D

Datenübertragungsgeschwindigkeit 67
 Delta-Sigma-A/D-Wandler 46
 Digitale Ausgänge 59
 Digitale Eingänge 59
 Softwarefunktionen 139
 Divisor-Faktor 50
 Download 36
 Dynamikbereich 19

E

Einbau der Karte 24
 Einbau der Karte (Foto) 25
 Elektromagnetische Verträglichkeit
 EMV 14
 EMV
 Elektromagnetische Verträglichkeit 14
 Energiebedarf 16
 Externe Clocks 60
 Externer Trigger
 Softwarefunktionen 149

F

FIR
 Finite Impulse Response 46
 Funktionen der Karte 41
 A/D-Wandler 48
 Analoge Ausgänge 52
 Modes 53
 Analoge Eingänge 42
 Anti-Aliasing-Filter 43
 Coupling Mode 43
 Eingangsbereich 43
 Kalibrierung 43
 Konfiguration 43
 Modes 51
 Verteilung 42
 Zyklischer Mode
 Ring Buffer 52
 Chronometer
 Modes der Chronometer 57
 Chronometer-Eingänge 56
 Datenübertragungsgeschwindigkeit 67
 Digitale Ausgänge 59
 Digitale Eingänge 59
 Externe Clocks 60
 Master Mode 61
 Slave Mode 62
 On Board-Speicher (SDRAM) 63
 Speicherkonzept
 Globaler Speicher 64
 Speicherkonzept
 Ring Buffer 64
 Stromquellen 50

G

Globaler Speicher 64
 Glossar 151
 Grenzwerte 15
 Analoge Ausgänge 20
 Analoge Eingänge 16
 Chronometer 21
 Digitale Ausgänge 21
 Digitale Eingänge 20
 Driver (Master board) 22
 Externer Clock 1/2 21
 Receiver (Slave board) 22
 Stromquellen 21

H

Handhabung der Karte 13

I

ICP-Sensor 50

K

Kartenkonfiguration mit ADDIREG 27

M

Master Mode 61
 Mechanischer Aufbau 14

N

Nyquist-Konzept 44

O

Offset-Fehler 20
 On Board-Speicher (SDRAM) 63

P

PC-Mindestvoraussetzungen 15
 PLD Clock 50

R

Registrieren einer neuen Karte 35
 Registrierung einer vorhandenen Karte ändern 35
 Registrierungsprogramm 27
 Ring Buffer 64
 APCI-3600 66
 Überblick 65

S

SDRAM 63
 On-Board-Speicher 63
 Sigma-Delta A/D-Wandler 48
 Signalgeneratormode (Free Run)
 Analoge Ausgänge 53
 Signalgeneratormode (Ring Buffer)
 Analoge Ausgänge 54
 Simple Mode
 Analoge Ausgänge 53
 Analoge Eingänge 51
 Chronometer 57
 Slave Mode 62
 Software 27
 Softwarefunktionen 69
 Allgemeine Funktionen 69
 Analoge Ausgänge 107
 Analoge Eingänge 90
 Chronometer-Modul 125
 Digitale Ausgänge 142
 Digitale Eingänge 139
 Speed Modes 48
 Steckerbelegung
 26-pol. Stecker auf 37-pol. SUB-D Stiftstecker 38
 Analoge Ausgänge, Chronometer-Eingänge 39
 Externe Clock 39
 Koaxialer SMB-Stiftstecker 37
 Stromquellen 50
 Anzahl 21
 Stromverbrauch 16

T

Technische Daten 14

U

Überspannungsschutz 18
 Update 36
 Software 36
 Urheberrecht 2

V

Version
 APCI-3600 40
 Versionen 15
 APCI-3600, APCI-3600-L 40

Z

Zubehör 14
 Zyklischer Mode: Auto Stop
 Analoge Eingänge 52
 Chronometer 57
 Zyklischer Mode: Ring Buffer
 Analoge Eingänge 52
 Chronometer 57